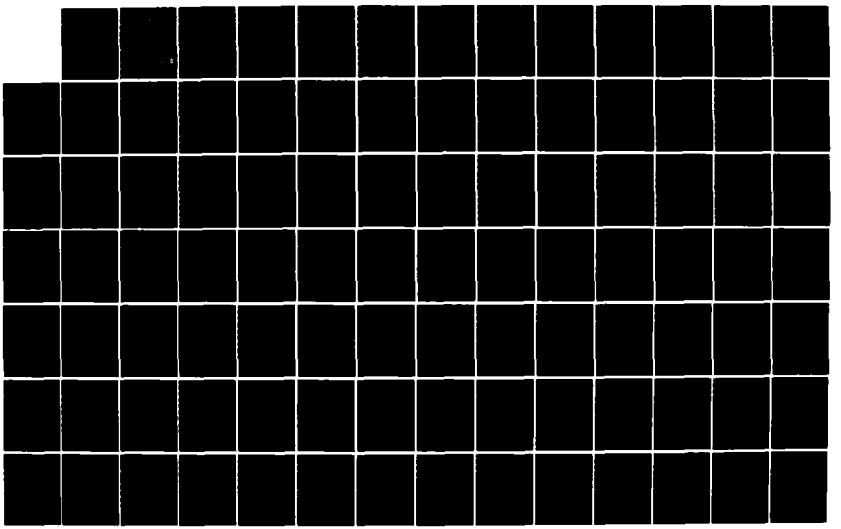
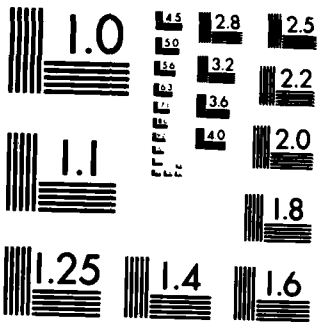
AD-A151 890 THE PERFORMANCE OF A NEW ADAPTIVE FILTER FOR DIGITAL 1/2
COMMUNICATIONS(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERING J R WAY
UNCLASSIFIED JUN 84 AFIT/GE/EE/84S-12 F/G 17/2 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

THE PERFORMANCE OF A NEW ADAPTIVE FILTER FOR DIGITAL COMMUNICATIONS

THESIS

AFIT/GE/EE/84S-12 John R. Way, Jr.
Capt USAF

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY (ATC)
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GE/EE/84S-12

THE PERFORMANCE OF
A NEW ADAPTIVE FILTER
FOR DIGITAL COMMUNICATIONS

THESIS

AFIT/GE/EE/84S-12 John R. Way, Jr.
Capt USAF

Approved for public release; distribution unlimited

AFIT/GE/EE/84S-84

THE PERFORMANCE OF A NEW ADAPTIVE FILTER FOR DIGITAL COMMUNICATIONS

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

by
John R. Way, Jr., M.S.S.M.
Capt USAF
Graduate Electrical Engineering
June 1984

Preface

This study resulted from the growing interest in the field of adaptive noise cancelling for digital communications, particularly spread spectrum systems. When my thesis advisor, Dr. V. H. Syed, suggested this topic, I welcomed it since I had recently become interested in spread spectrum communications and I envisioned spending half my time on spread spectrum and the other half on adaptive filters. I didn't realize then how much I would learn about computer programing, simulation, and modeling. In fact, it appears that a major product of this thesis effort is a computer model of a spread spectrum receiver preceded by a particular adaptive filter. I hope that this computer program can be used as a starting point for future theses of this type.

It is a pleasure to acknowledge my indebtedness to Dr. Syed without whose advice and guidance this project would not have been possible. He patiently answered even my most basic questions which were many since I have never taken a digital signal processing or spectral estimation course. Dr. Syed's ability to provide information was amazing and his assistance with technical problems cannot be praised enough. His constant good humor and enthusiasm helped me overcome my many frustrations.

I also wish to thank Major Castor for taking the time to read my draft and for his valuable comments. In addition,

special thanks are due Major Castor for his support, encouragement, and confidence in me when I was falling behind.

I would also like to express my appreciation to Captain Lawlis of the Mathematics Department whose help with some of the computer programming problems saved me countless hours of anguish.

Finally, I would like to express my gratitude to my wife, Gayle, for her encouragement, patience, sacrifice, and understanding not only during this research effort but during the entire academic program. Without her, this project would have been much more frustrating and much less rewarding.

John R. Way, Jr.

Contents

List of Figures

Abstract

The improvement in performance of a digital communications receiver preceded by an adaptive filter using the soft-constraint Least Mean Square algorithm is found. This algorithm has previously only been used in conjunction with adaptive antenna arrays. The fundamentals of adaptive filters and the spread spectrum technique are presented. A computer model is developed which simulates a 19 tap adaptive filter and spread spectrum receiver combination in the baseband region. The filter is shown to adapt well to hopping jammers and dual jammers. Plots of the filter's transfer function show notch depths ranging from -34 dB to -77 dB over iterations ranging from 15 to 5000 for the case of a 20 db sinusoid jammer. The improvement gained by adjusting the softness of the constraints is presented. The computer program and recommendations for further study are included as well as a 44 item bibliography.

THE PERFORMANCE OF A NEW ADAPTIVE FILTER FOR DIGITAL COMMUNICATIONS

I. INTRODUCTION

Background

As recently as April 1983 (Ref. 1), the Director of the Joint Electronic Warfare Center, Major General Doyle E. Larsen, USAF, wrote, "The criticality of good communications to military operations cannot be overemphasized. The commander depends upon intelligence transmitted by communications systems to formulate his plan...Despite the known Soviet jamming threat, none of our military services today *posess adequate jam resistant, secure communications.* In every recent exercise involving electronic warfare...lack of communications has been cited as the glaring deficiency. To solve this problem, secure, jam resistant communications must be established..." To accomplish this, the military is relying increasingly on spread spectrum (SS) systems because of their inherent antijam, anti-intercept, and anti-interference properties.

In fact, an SS system known as HAVE QUICK is being used in the F-15, F-16, and F-111 jets (Refs. 2:51; 3:57). Another SS system called the Joint Tactical Information Distribution System (JTIDS) has been tested in the E-3A, in a helicopter, in cargo aircraft, and at ground-based centers

(Ref. 4:105).

A military satellite, DSCS-II (Defense Satellite Communications System), is also known to use an SS system (Ref. 5:80) and a new satellite, called Mil-Star (military strategic-tactical and relay), emphasizing SS techniques could become operational before 1990 (Ref. 6:72).

The Space Shuttle will also use SS for navigation in conjunction with the Global Positioning System (GPS) satellites (Ref. 7:54.4.1). In addition, civilians may get to use SS systems in mobile radio communications if recent proposals are accepted (Refs. 8:G3.3.1; 9:855).

However, due to system bandwidth constraints and/or a hostile environment, the SS system's inherent anti-interference property may not sufficiently reduce the effects of narrowband interference. Therefore, some method of reducing a jammer's interference could be used in conjunction with an SS system to produce a highly jam resistant communication system. One method to reduce interference is to use an adaptive notch filter.

A filter may be described as "any device or system that processes incoming signals or other data in such a way as to smooth or classify them, predict future values, or eliminate interference. Adaptive filters are devices that automatically adjust their own parameters and seek to optimize their performance according to a specific criterion. Though somewhat more difficult to design, analyze, and build than fixed filters, they offer the potential of substantial

improvements in performance when signal properties are unknown or variable with time" (Ref. 10:1143). Thus, adaptive filters are very attractive for use in interference rejection since they can adapt to any frequency the jammer choses.

In fact, there is a considerable amount of interest in the use of adaptive filters with SS systems as evidenced by the large volume of literature in this area (for example, refs. 11, 12, 13, 14, 15:570-579).

Scope

Many different adaptive filters are available and new algorithms which characterize these adaptive filters are frequently being introduced. After reviewing many new adaptive filter algorithms (see, for example, refs. 16, 17, 18, 19), an algorithm derived by Widrow, Chestek, Mesiwab, and Duvall, and applied to adaptive antenna arrays was selected for this study (Ref. 20). The algorithm is applied to an adaptive filter and the filter and an SS receiver are modeled by a computer program in the baseband region.

Problem

This study determines the improvement in performance gained by using the above adaptive filter in conjunction with an SS receiver. This will be accomplished by comparing the signal-to-jammer ratios at the output of the receiver with and without the adaptive filter.

Assumptions

The following assumptions are made to narrow the problem and simplify the calculations:

1. The noise introduced in the communication channel is additive white Gaussian noise (AWGN). This is the most destructive type of additive noise (Ref. 21:6).
2. The jammer's signal is modeled as a continuous wave (CW), i.e., a single frequency sinusoid.
3. Synchronization of the psuedonoise (PN) code in the SS receiver is not considered a problem.
4. In the simulation, carrier recovery is considered perfect. Thus, the effect of jamming on the circuitry regenerating the carrier is ignored (Ref. 22:15.6.2).
5. The adaptive filter and SS receiver are modeled in the baseband region and the SS signal pulse width is normalized to one second. It is assumed that when the appropriate frequency and time components are scaled to real world values the significant results of this thesis effort will not change.

Approach and Organization

Closed-form analysis of spread spectrum systems are "useful for preliminary design guidance and insight into gross link behavior...Because these systems are coherent and require strict synchronism between receiver and transmitter,

mathematical complexities may render closed-form techniques of little value to those who must deal with link functioning under strongly stressed, jammed conditions. In these situations, computer simulation is the only viable technique available, short of actual system bench experiments" (Ref. 22:15.6.1). Therefore, all final results are from computer simulation and not from analytical calculations.

Thus, following this introductory chapter, presented in Chapter II is a review of SS principles and the SS system model. Chapter III contains a description of the adaptive filter and a development of its algorithm. Chapter IV explains the computer model and the results are presented in Chapter V. Finally, in Chapter VI, conclusions of this study are drawn and recommendations for future work on this subject are suggested.

II. SPREAD SPECTRUM FUNDAMENTALS

The spread spectrum (SS) technique can be considered as a second stage of signal processing, beyond the primary stage where the information bearing signal modulates the carrier signal (Ref. 23:85). A good definition of SS is as follows (Ref. 9:855):

"Spread Spectrum is a means of transmission in which the signal occupies a bandwidth in excess of the minimum necessary to send the information; the band spread is accomplished by means of a code which is independent of the data, and a synchronized reception with the code at the receiver is used for despreading and subsequent data recovery."

Although this definition excludes chirped spread spectrum systems, there are three key points that should be expanded and emphasized. First, the spread bandwidth is greatly in excess of the minimum required, on the order of 10^3 to 10^6 times the minimum (Refs. 5:78; 24:11; 25:29).

Second, the code used to spread the bandwidth is a long psuedorandom or psuedonoise (PN) binary-valued sequence (Ref. 15:546). These code sequences are designated as psuedonoise codes because they make the signals appear somewhat noiselike if intercepted by a receiver which does not have access to the exact code sequence (Ref. 26:10).

Finally, the intended receiver must know the code

sequence exactly or have a method of generating it himself before reception of the information can be accomplished.

Advantages and Disadvantages

Some of the advantages of using an SS system include:

1. Antijamming
2. Anti-interference
3. Code Division Multiple Access (CDMA)
4. Low Probability of Intercept (LPI)
5. Message Privacy.

The first three advantages occur because of the inherent interference rejection capability of SS systems.

Antijamming. By far the most important and common use of an SS system is to prevent an enemy jammer from disrupting military communications (Refs. 3, 4, 14, 23, 27, 28). For example, an enemy jammer would need a power output of one million watts with a 28-ft diameter antenna to disrupt a military satellite using spread spectrum operating at Extremely High Frequency (EHF). "Achieving 1 megawatt of power at EHF would be difficult despite Soviet advances in 'gyrotron' microwave tubes. It would be possible to use a lower transmitter [power] at EHF with a larger antenna, but it is very difficult to build a much larger antenna with the precision tolerences required...Generating very high levels of power...also is both difficult and costly" (Ref. 6:73).

Anti-interference. Just as the SS system suppresses intentional interference (jamming), it also suppresses self-interference caused by multipath in which delayed

versions of the signal, arriving via alternate paths, interfere with the direct path transmission (Ref. 24:12).

Code Division Multiple Access. By assigning appropriately different PN code sequences to different transmitter-receiver pairs, the users can communicate at the same time and over the same channel bandwidth with limited interference. This efficient utilization of the RF spectrum is called code division multiple access (CDMA) or spread spectrum multiple access (SSMA) (Refs. 29:21; 30:720).

Low Probability of Intercept. A message may be hidden in the background noise by spreading its bandwidth and transmitting at a low average power (Ref. 15:545). If the SS power density is less than that of atmospheric, galactic, and receiver front end noise, the SS signal will not even be detected (Ref. 5:78). For example, it is not known whether bootleg radio amateurs are using SS modulation to evade FCC regulations (Ref. 24:12).

Message Privacy. Private messages may be sent because only the intended receivers know the psuedorandom code sequence necessary to despread the signals and recover the messages (Ref. 15:545).

Additional advantages of SS systems occur outside the area of communications. These include accurate range (time delay) and range rate (velocity) measurements in radar and navigation (Ref. 15:544).

The main disadvantage of an SS system is complexity. An SS system must include code sequence generators, correlators

or matched filters, code tracking loops and other subsystems not required in a more conventional system.

Types of SS Systems

The means by which the spectrum is spread is determined by the type of the SS system. Most types are characterized by one of the following:

1. Direct-spread or direct-sequence (DS).
2. Frequency-hopping (FH).
3. Time-hopping (TH).
4. Chirp Technique.
5. Hybrid Systems.

Direct-sequence. Spread spectrum systems using DS modulation appear to be the most common or at least the most well known. As the name implies, the PN code sequence directly modulates the data signal to spread the bandwidth. This study is concerned exclusively with the direct-sequence SS system, sometimes referred to as a psuedonoise SS system.

Frequency-hopping. In frequency-hopping SS systems a narrowband information bearing signal is psuedorandomly jumped across a wide range of frequencies. The PN code sequence is used to determine where in a wide bandwidth the narrow-band signal is transmitted. "Since the receiver has knowledge of this sequence, a narrow-band filter can be used to pass the signal power with minimum loss while rejecting unwanted jammer power" (Ref. 23:87).

Time-hopping. In time-hopping SS systems, the PN code sequence is used to key the transmitter on and off. Thus, TH

is really just pulse modulation and is similar to time division multiplexing. In actual practice, TH is usually used in a hybrid form with FH (Ref. 31:38-39).

Chirp Technique. Chirp systems are spread spectrum systems that do not normally employ a psuedorandom code but do use a wider bandwidth than that absolutely required (Ref. 31:40). "A chirp signal is generated by sliding the carrier over a given range of frequencies in a linear or some other known manner during a fixed pulse period. This results in pulse-FM signal whose bandwidth is limited only by physical ability to shift a carrier frequency and by the ability to construct a receiver to demodulate it.

"The idea behind chirp signals is that the receiver can employ a matched filter of a relatively simple design to reassemble the time-dispersed carrier power in such a way that it adds coherently and thus provides an improvement in signal to noise" (Ref. 32:22). The chirp technique is used almost exclusively in radar applications (Ref. 33:135, 136, 143, 149).

Hybrid Systems. "In addition to the more usual forms of spread spectrum modulation, there are the hybrid combinations of modulation which offer certain advantages over, or at least extend the usefulness of, the direct sequence and frequency hopping techniques. The most often used combination (or hybrid) spread spectrum signals are made up of (a) simultaneous frequency hopping and direct sequence modulations, (b) simultaneous time and frequency hopping, and

(c) simultaneous time hopping and direct sequence modulations" (Ref. 31:44).

Spread Spectrum System Model

A model of a binary phase shift keyed (BPSK) SS communications system using DS modulation is shown in Figure II-1 (Ref. 34:11). The baseline components of this model are:

- m(t) is the digital message sequence (±1).
- PN(t) is the PN code sequence (±1).
- w_o is the carrier frequency (rad/sec).
- Tm is the length of a message bit (sec).
- E/Tm is the signal power (watts).

The bipolar binary message sequence is first multiplied by the carrier which results in a phase shift of zero or Π radians. Then the phase modulated carrier is multiplied by the PN code sequence, PN(t), and sent to the receiver through the channel.

The receiver uses its own identical PN code to strip the PN code off the received signal as illustrated in Figure II-2. The received signal, r(t), is multiplied by the synchronized reference PN code to obtain the phase modulated carrier. This phase modulated carrier is then demodulated by conventional means such as a matched filter or a correlation detector followed by a decision device. It is easy to see that the main difference between SS systems and conventional systems is the mutliplication by the PN code sequence at the transmitter and receiver.

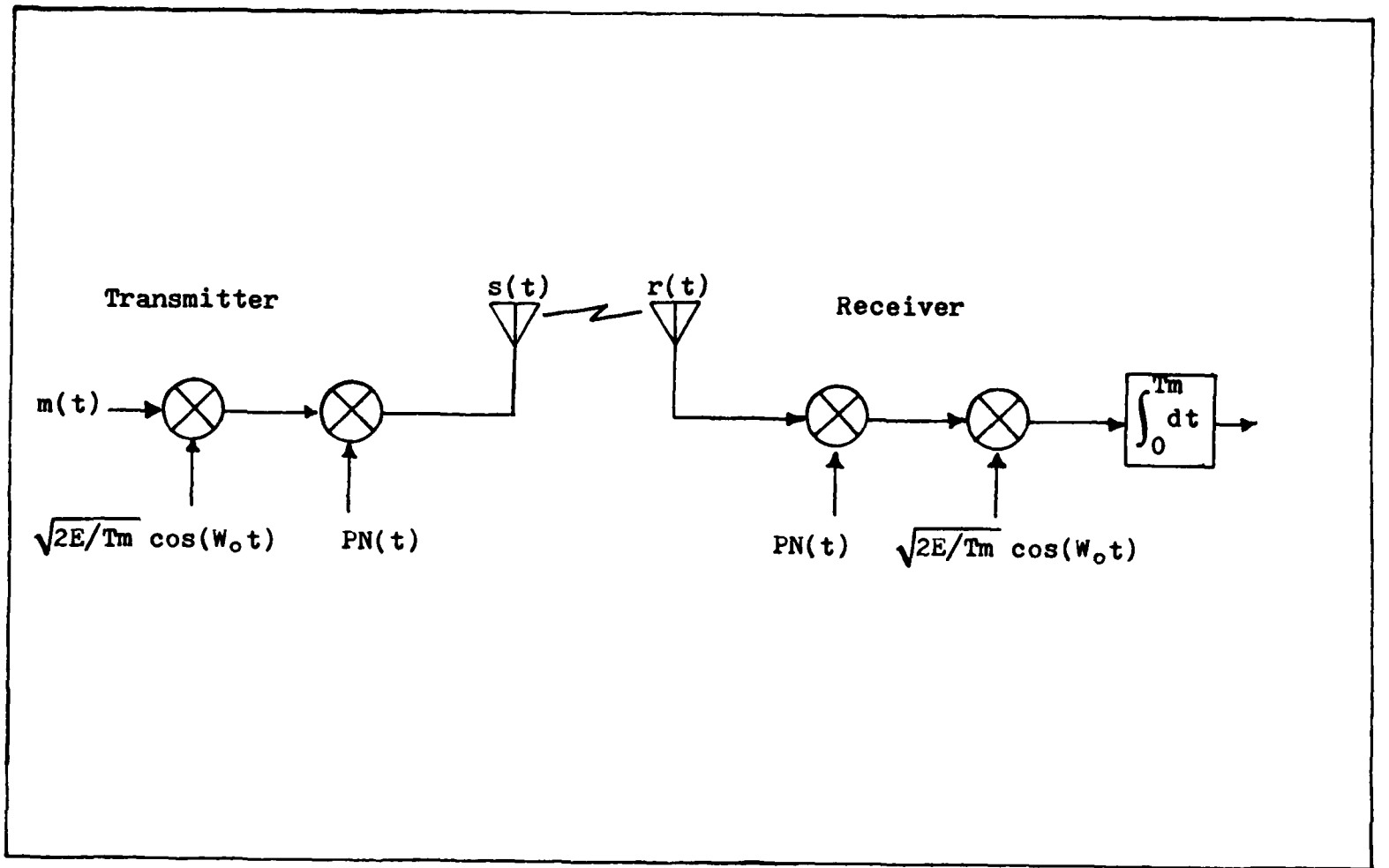


Figure II-1 Spread Spectrum Communications Systems Model (Ref. 37:8)

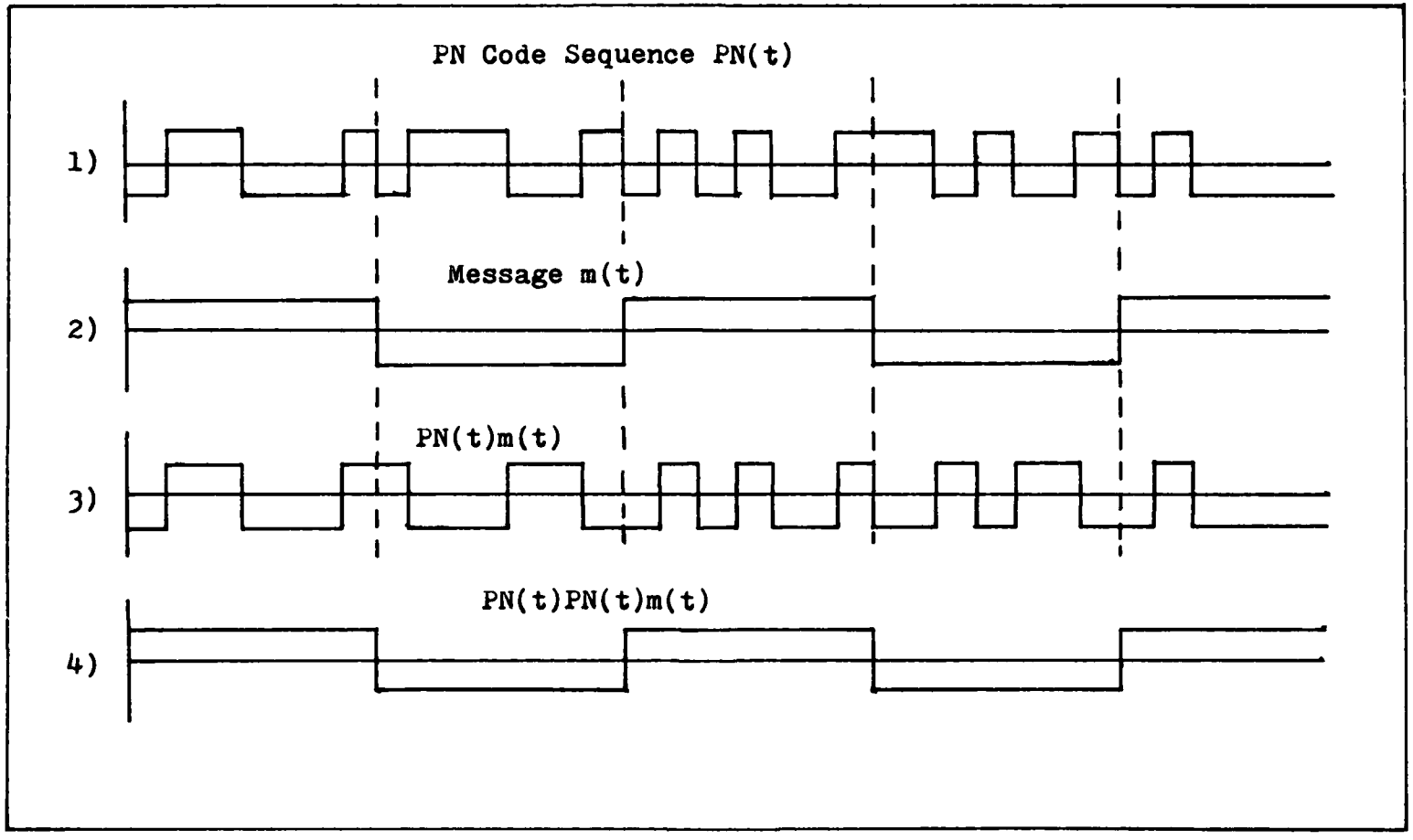


Figure II-2 Message Modulation/Demodulation of a PN Sequence (Ref. 34:14)

PN Code Sequence

The pseudorandom or pseudonoise (PN) code sequence is a periodic series of bits or "chips" of length Tc seconds and amplitude ± 1 (Ref. 34:15). As noted earlier, these sequences are designated as pseudonoise codes because they make the signals appear somewhat noiselike if intercepted by a receiver which does not have the exact code sequence (Ref. 26:10).In addition, the rate of the PN code is typically one thousand times the rate of the message bits (Refs. 26:8; 28:23; 29:26). As a result of the modulation of the message signal by the PN code sequence, the power in the transmitted signal is spread in frequency over a bandwidth corresponding to the PN clock frequency. Thus, because the transmitted energy is spread over a wide bandwidth, the SNR at the receiver input is low. This contributes to the signal appearing as wideband noise to the unintended receiver.

To simplify the analysis, some of the properties that must be assumed about the PN code sequence are (Ref. 34:15):

- the sequence is continuous, i.e., no spaces between bits.
- the positive and negative chips occur with equal probability.
- successive pulses are statistically independent.

An arbitrary portion of a typical PN sequence is illustrated in Figure II-3a.

Papoulis (Ref. 35:294) shows that the autocorrelation function of the PN code sequence, $R_{pn}(\tau)$, is given by

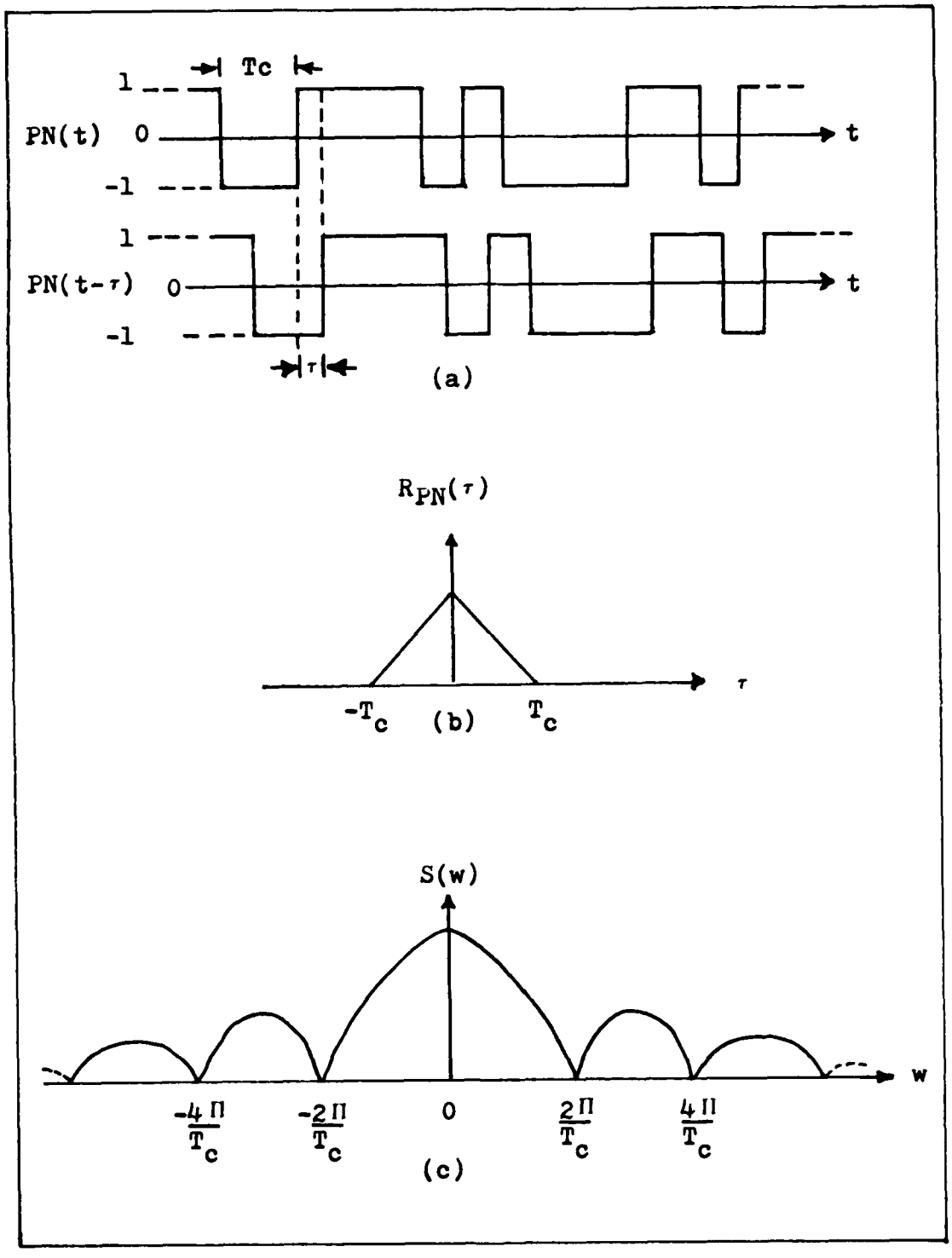


Figure II-3 (a) PN Code Sequence (b) Autocorrelation Function (c) Power Spectral Density (Ref. 34:16)

$$Rpn(\tau) = \begin{cases} 1 - |\tau|/Tc & \text{for } |\tau| < Tc \\ 0 & \text{for } |\tau| > Tc \end{cases} \quad (1)$$

as illustrated in Figure II-3b.

The power spectral density (PSD), Spn(w), for the PN code sequence is found by taking the Fourier transform of the autocorrelation function (Ref. 35:42) to obtain:

$$Spn(w) = Tc \sin^2(wTc/2)/(wTc/2) \quad (2)$$

The PSD, Spn(w), is illustrated in Figure II-3c.

Performance of the Spread Spectrum System

The transmitted signal, s(t), of the SS system model in Figure II-1 is

$$s(t) = \sqrt{2E/Tm}\, m(t)PN(t)\cos(w_o t) \quad (3)$$

The message bit interval, Tm, is an integer multiple of Tc, the code bit interval, in practical SS systems (Ref. 15:548). The signal at the receiver is given by

$$r(t) = s(t) + n(t) \quad (4)$$

where n(t) is additive white Gaussian noise (AWGN) with zero mean. The noise is independent of the PN code and has a PSD in watts/Hz given by

$$Sn(w) = No/2 \quad (5)$$

Synchronization of the PN code in the receiver will not be addressed in this thesis. However, it must be noted that

synchronization is critical for correct correlation with the PN code. After synchronization, the received signal, r(t), is multiplied by a stored replica of the PN code to become

$$r(t) = \sqrt{2E/Tm}\, m(t)PN^2(t)\cos(w_o t) + PN(t)n(t) \qquad (6)$$

Let

$$z(t) = PN(t)n(t) \qquad (7)$$

Now, since $PN^2(t)$ always equals one, r(t) becomes

$$r(t) = \sqrt{2E/Tm}\, m(t)\cos(w_o t) + z(t) \qquad (8)$$

The autocorrelation of z(t) is given by

$$\begin{aligned} Rz(t1,t2) &= E[z(t1)z(t2)] \\ &= E[PN(t1)n(t1)PN(t2)n(t2)] \\ &= E[PN(t1)PN(t2)]E[n(t1)n(t2)] \\ &= Rpn(t1-t2)(No/2)\,\delta(t1-t2) \end{aligned} \qquad (9)$$

Since the delta function is zero except when t1=t2, the autocorrelation simplifies to

$$\begin{aligned} Rz(t1-t2) &= Rpn(0)(No/2)\,\delta(t1-t2) \\ &= (No/2)\,\delta(t1-t2) \end{aligned} \qquad (10)$$

Thus, z(t) is also white Gaussian noise (WGN) and the problem reduces to a threshold detection problem in a system using binary antipodal signaling in WGN (Ref. 34:19). Therefore, assuming the message bits are equally probable, the probability of error P(E) is given by

$$P(E) = \text{erfc}(\sqrt{2E/No}) \tag{11}$$

where erfc(x) is the complementary error function, defined as

$$\text{erfc}(x) = 1/\sqrt{2\pi} \int_x^\infty \exp(-y^2/2)dy \tag{12}$$

Therefore, the performance of an SS system in AWGN is identical to a conventional binary antipodal signaling system (Refs. 34:20; 15:146).

Performance of SS System with Jammer (Refs. 34, 37)

The performance advantage of SS systems become apparent when studied in the presence of a narrowband jammer transmitting in the spread bandwidth.

First, assume the jammer's signal, j(t), to be Gaussian with zero mean and PSD, Sj(w), as shown in Figure II-4. The jammer's signal must also be uncorrelated with the PN code. The signal that arrives at the receiver now becomes

$$\begin{aligned} r(t) &= s(t) + n(t) + j(t) \\ &= \sqrt{2E/Tm}\, m(t)PN(t)\cos(w_o t) + n(t) + j(t) \end{aligned} \tag{13}$$

As before, the received signal is multiplied by the PN code sequence and becomes

$$r(t) = \sqrt{2E/Tm}\, m(t)PN^2(t)\cos(w_o t) + PN(t)[n(t) + j(t)] \tag{14}$$

Let $z(t) = PN(t)[n(t) + j(t)]$. Now r(t) is given by

$$r(t) = \sqrt{2E/Tm}\, m(t)\cos(w_o t) + z(t) \tag{15}$$

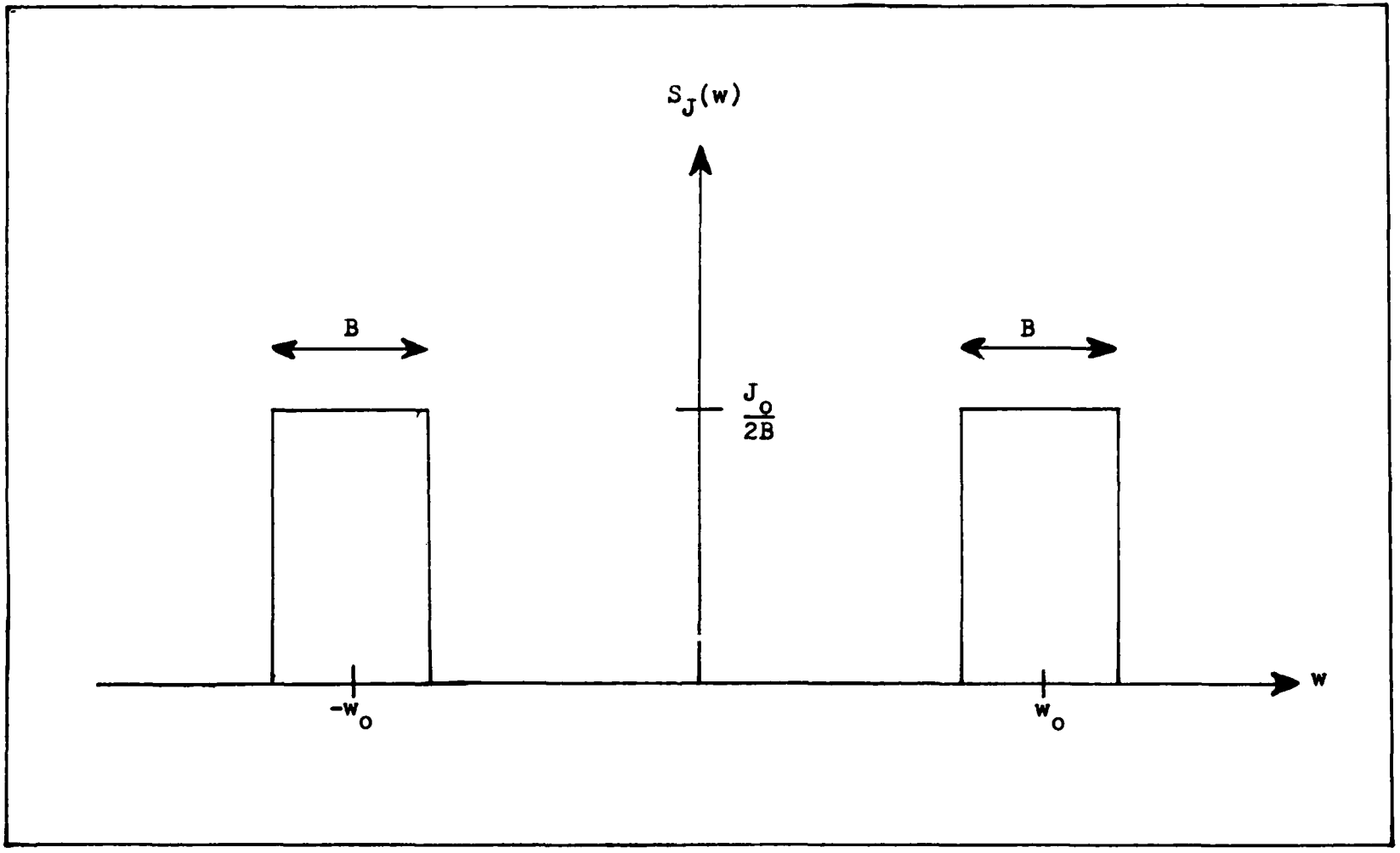


Figure II-4 Power Spectral Density of Jamming Signal (Ref. 37:16)

Thus, the PN code is removed from the message signal and the jammer's signal is reduced as shown qualitatively in Figure II-5. The relative spectra of the received SS signal and jammer are shown in Figure II-5a. Figure II-5b shows the relative spectra after multiplication by the PN code in the receiver. The message signal is despread from the PN code bandwidth, Bpn to the original message bandwidth Bm while the jammer's signal is spread over a bandwidth exceeding *Bpn*.

The autocorrelation of z(t) is described by

$$\begin{aligned} Rz(t1,t2) &= E[z(t1)z(t2)] \\ &= E[PN(t1)PN(t2)]E[(n(t1) + j(t1))(n(t2) + j(t2))] \\ &= Rpn(t1-t2)[E[n(t1)n(t2)] + E[n(t1)j(t2)] \\ &\quad + E[n(t2)j(t1)] + E[j(t1)j(t2)]] \end{aligned} \tag{16}$$

Since n(t) and j(t) are independent the autocorrelation can be rewritten as

$$\begin{aligned} Rz(t1,t2) &= Rpn(t1-t2)[Rn(t1-t2) + Rj(t1-t2)] \\ &= Rpn(t1-t2)[(No/2)\,\delta(t1-t2) + Rj(t1-t2)] \end{aligned} \tag{17}$$

The autocorrelation of the jamming signal, $Rj(\tau)$, is found by taking the inverse Fourier transform of Sj(w) = Jo/2B with τ = t1-t2 and integrating between Wo-B/2 and Wo+B/2. Using the above, Shepard (Ref. 34:23-24) then shows that the autocorrelation of z(t) becomes

$$Rz(\tau) = No/2\,\delta(\tau) + Rpn(\tau)JoCOS(Wo)[SIN(B\pi\tau)]/(B\pi\tau) \tag{18}$$

For the case of narrowband jamming, B approaches zero; thus

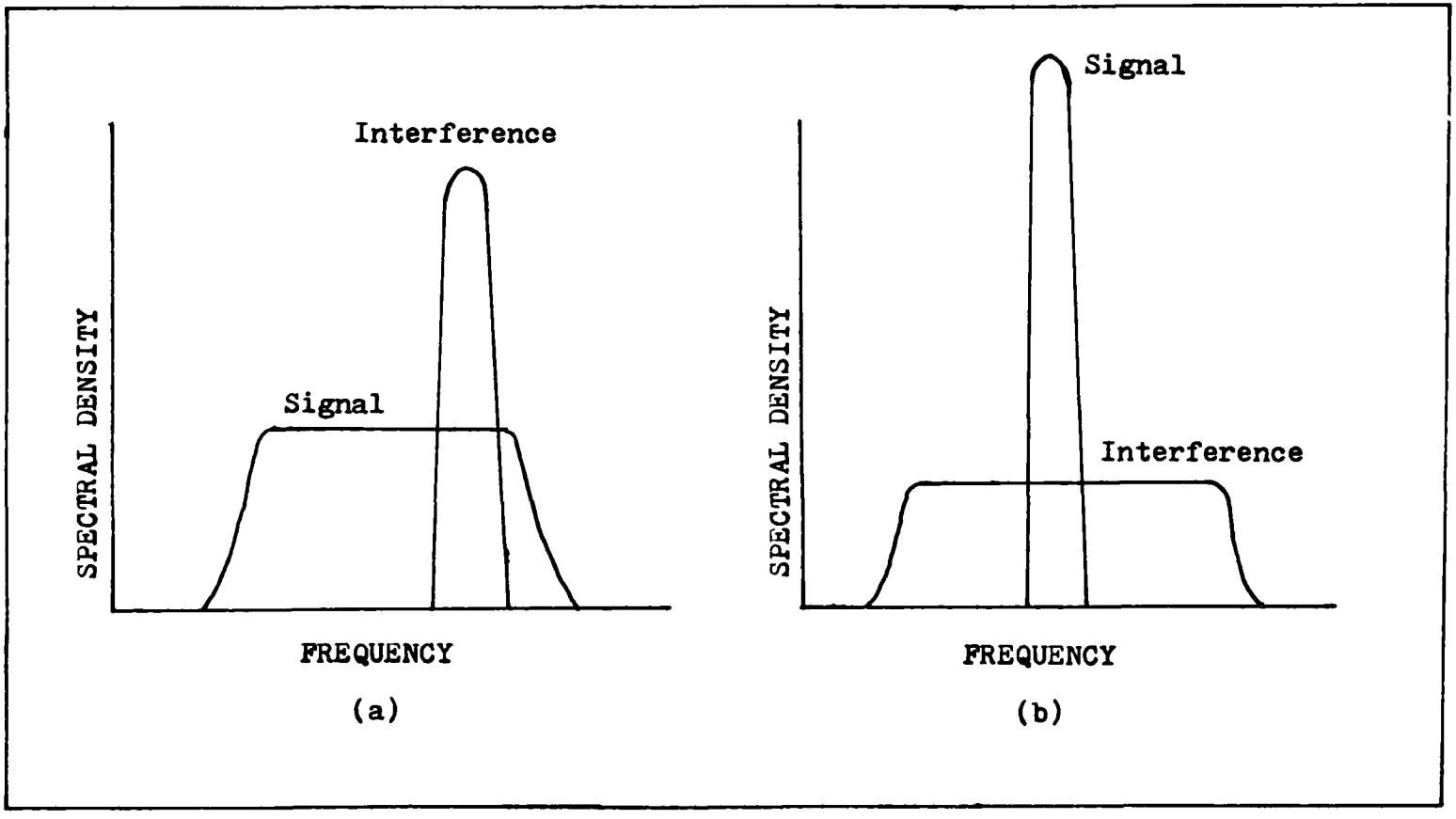


Figure II-5 Spectra of Desired Signal and Jammer (a) Received Signal (b) Correlation Detector Output (Ref. 34:22; 21:27)

$Rz(\tau)$ reduces to

$$Rz(\tau) = (No/2)\,\delta(\tau) + Rpn(\tau)Jo[\cos(Wo\tau)] \tag{19}$$

and the PSD of z(t) becomes

$$Sz(w) = No/2 + (1/2)JoTc\frac{SIN^2[(W-Wo)Tc/2]}{[(W-Wo)Tc/2]} + (1/2)JoTc\frac{SIN^2[(W-Wo)Tc/2]}{[(W-Wo)Tc/2]} \tag{20}$$

Figure II-6 shows the PSD, Sz(w). Since the message pulse duration, Tm, is generally several orders of magnitude greater than the PN code pulse duration, Tc, the bandwidth of PN(t)j(t) is much wider than the bandwidth of PN(t)s(t). The PSD of PN(t)j(t) is approximately flat over the message bandwidth (see Figure II-5b) and is processed by the correlation detector as AWGN. Thus, using this approximation the PSD of z(t) is

$$Sz(w) = No/2 + JoTc/2 \tag{21}$$

and the probability of error becomes

$$P(E) = \text{erfc}[\sqrt{2E/(No + JoTc)}] \tag{22}$$

To see the improvement in performance by using SS techniques, the P(E) above should be compared to the case when no PN code is used, or equivently, when PN(t) always equals one. For this case, Shepard (Ref. 34:25-28) and Hall (Ref. 37) show that

$$P(E) = \text{erfc}[\sqrt{2E/(No + JoTm)}] \tag{23}$$

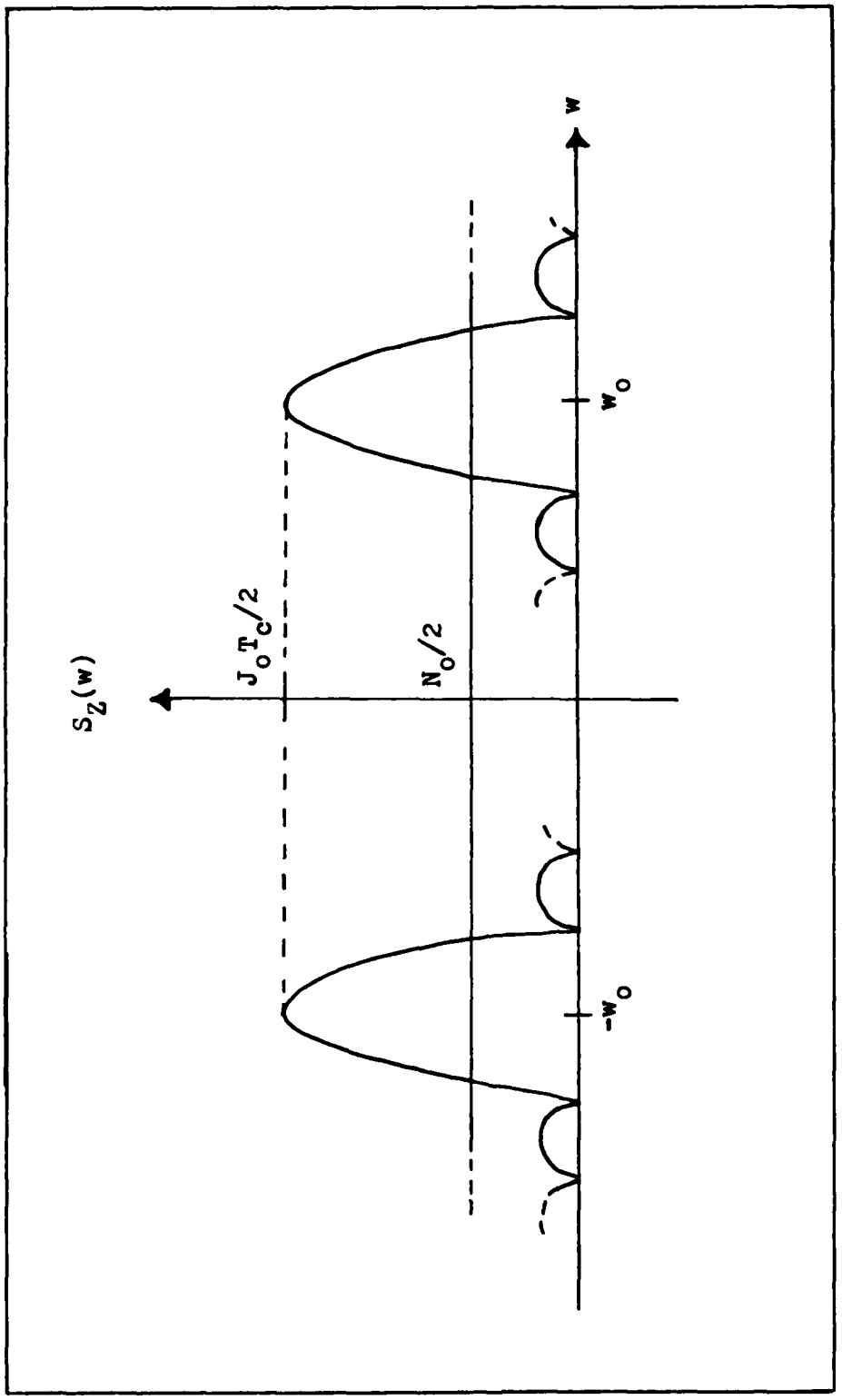


Figure II-6 Power Spectral Density of z(t) (Ref. 37:18)

Equations (22) and (23) can now be compared to see the effect of the code, PN(t), on system performance. Observe that if the noise is considered negligible, which can generally be done for the case of a strong jamming signal, then the effect of the narrowband jammer on the system performance was reduced from JoTm to JoTc.

The improvement gained through the use of a PN code can be expressed as the ratio

$$(JoTm)/(JoTc) = Tm/Tc = N \tag{24}$$

Processing Gain

The improvement ratio derived above is commonly referred to as the processing gain (PG) of the SS system and it represents the advantage gained over the jammer that is obtained by expanding the bandwidth of the transmitted signal (Ref. 15:556). Since the RF transmitted bandwidth equals 1/Tc, and the data (message) bandwidth equals 1/Tm, the processing gain can equivently expressed as

$$PG = Tm/Tc = Bpn/Bm \tag{25}$$

Another way of defining the PG is as

$$(S/N)o = PG(S/N)i \tag{26}$$

where

(S/N)o = signal to noise ratio out of the receiver

(S/N)i = signal to noise ratio into the receiver

Observe that it appears that an SS receiver can perform satisfactorily when faced with a jammer (or noise) signal having a power level larger than the desired signal by the amount of available processin gain (Ref. 28:7). However, in reality, the requirement for a useful signal to noise system output as well as internal losses must be taken into account. The jamming margin, Mj, which might be attained is given by

$$Mj = PG - L - (S/N)o \tag{27}$$

where

PG = processing gain in dB

L = the system implementation loss in dB

(S/N)o = the operationally required ratio in dB at the information output (Ref. 26:26).

For example, suppose that the SS bandwidth, Bpn, is 1000 times that of the message bandwidth, Bm, so that PG = 30 dB. Also, suppose that for an acceptable probability of error the (S/N)o is required to be 10 dB and L is taken to be 2 dB. Now, the jamming margin becomes

$$Mj = 30 - 2 - 10 = 18 \text{ dB}$$

Therefore, for this example, the jamming power could not exceed the message signal power by more than 18 dB and still maintain the desired performance (Ref. 29:26).

The spread spectrum fundamentals, communications system model, and performance analysis presented in this chapter should sufficiently prepare the reader to understand the

basis for the computer simulation and appreciate the SS system's inherent interference rejection capability. The next chapter will discuss the adaptive filter's role in interference rejection.

III. ADAPTIVE FILTERS

Background

The usual method of estimating a signal which is degraded by additive noise is to pass it through a filter that is designed to suppress the noise while leaving the signal relatively undisturbed. When there is prior knowledge of both the signal and noise a fixed filter is suitable for noise suppression (Ref. 39:1692-1693). However, when no prior knowledge of the noise is known as in the case of an enemy jammer, then an adaptive filter should be used for noise (jammer) suppression.

An adaptive filter, in its general form, is a device that adjusts its internal parameters and optimizes its performance according to the statistical characteristics of its input and output signals (Ref. 40:616). The adjustment of its internal parameters is accomplished in a single process by a recursive algorithm that automatically updates the system's parameters with the arrival of each new data sample (Ref. 41:563).

Adaptive Linear Combiner

The heart of almost all adaptive filters is the adaptive linear combiner shown in Figure III-1. In the adaptive linear combiner a set of n measurements, X1, X2, ... , Xn, is sampled to form n sampled measurements at some time, t. Each measurement, Xi(t), is multiplied by a corresponding

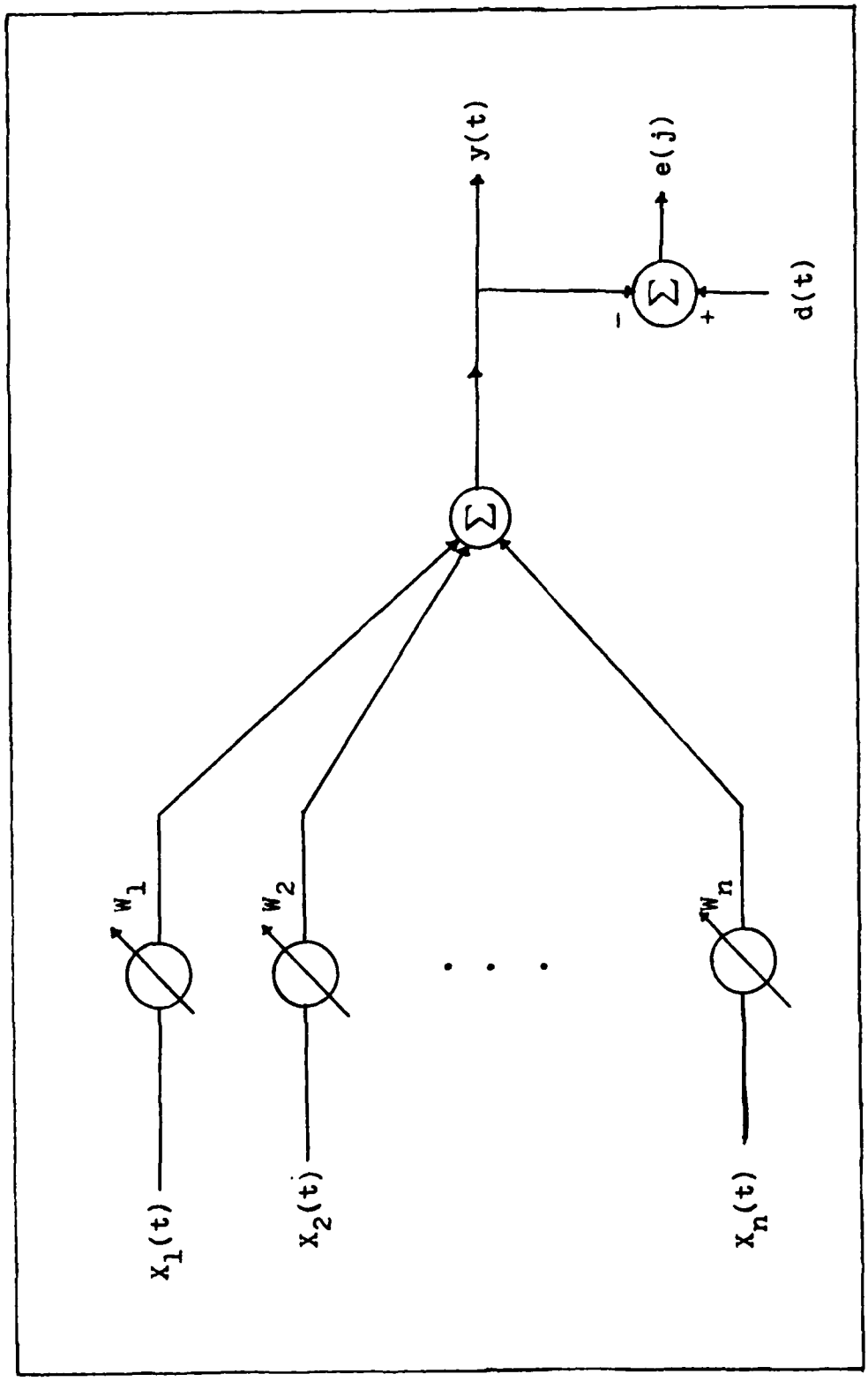


Figure III-1 Adaptive Linear Combiner (Ref. 10:114)

weighting function, Wi, and the weighted measurements are summed to form an output, y(t) (Ref. 10:1143). Thus, y(t) is given by (Ref. 42:13)

$$y(t) = W1X1(t) + W2X2(t) + \ldots + WnXn(t) \tag{28}$$

Let X(t) be a vector such that

$$X(t) = \begin{bmatrix} X1(t) \\ X2(t) \\ \vdots \\ Xn(t) \end{bmatrix} \tag{29}$$

and W be a vector such that

$$W = \begin{bmatrix} W1 \\ W2 \\ \vdots \\ Wn \end{bmatrix} \tag{30}$$

Then y(t) can be written as (Ref. 42:14)

$$y(t) = X^T(t)W = W^TX(t) \tag{31}$$

This output, y(t), is compared with a desired response, d(t), to form an error signal, e(t). The objective is to choose the weighting coefficients in such a way as to minimize the error signal and find the weighted sum of the input signals that is closest to the desired response (Ref. 10:1143). Note that as e(t) approaches zero, the output, y(t), approaches the desired response, d(t), since e(t) = d(t) - y(t).

We wish to find an optimal value Wopt of the weight vector W such that the mean square value of e(t) is minimized (Ref. 42:15). That is, Wopt must minimize

$$E[e^2(j)] = E[(d(t) - y(t))^2] \tag{32}$$

To facilitate computation via a digital computer, allow sampling with a period, T (Ref. 42:15). Then y(t) becomes

$$y(jT) = X^T(jT)W \; ; \qquad j = 0, 1, 2, \ldots \tag{33}$$

Or, simply

$$y(j) = X^T(j)W = W^TX(j) \tag{34}$$

Similarly, e(t) becomes

$$\begin{aligned} e(j) &= d(j) - y(j) \\ &= d(j) - W^TX(j) \end{aligned} \tag{35}$$

Therefore,

$$\begin{aligned} E[e^2(j)] &= E[(d(j) - W^TX(j))^2] \\ &= E[(d(j) - W^TX(j))(d(j) - W^TX(j))] \end{aligned} \tag{36}$$

After expanding terms, the equation becomes

$$\begin{aligned} &E[d(j)d(j) - 2d(j)W^TX(j) + W^TX(j)X^T(j)W] \\ &= E[d^2(j)] - 2W^TRdx + W^TRxxW \end{aligned} \tag{37}$$

where

$$Rxx = E[X(j)X^T(j)] \tag{38}$$

is the correlation matrix of X(j), and

$$Rdx = E[d(j)X(j)] \tag{39}$$

is the cross correlation vector between the input signals, Xi(j), and the desired response, d(j) (Refs. 21:212; 42:16).

To minimize equation (37), take the gradient, ∇, by differentiating the equation with respect to the weight vector, W, and set the result equal to zero:

$$\nabla[E(d^2(j)) - 2W^TRdx + W^TRxxW]$$
$$= -2Rdx + 2RxxW = 0 \tag{40}$$

Thus the desired least mean square (LMS) solution for the optimal weight vector, Wopt, is given by

$$Wopt = Rxx^{-1}*Rdx \tag{41}$$

Equation (41) is a matrix form of the Wiener-Hopf equation for the minimum mean square error filtering problem (Refs. 10:1144; 42:17).

Although the above equation gives the optimal weight vector for the adaptive linear combiner, an exact solution would require a priori knowledge of the correlation matrices Rxx and Rdx. However, an approximate solution can be found by using numerical means. Unfortunately, this method would require the inversion of an n X n matrix and as many as n(n + 1)/2 autocorrelation and cross correlation measurements to obtain the elements of Rxx and Rdx (Ref. 10:1144).

Another approximate solution to equation (41) uses a

relatively simple iterative procedure called the Steepest Descent LMS algorithm and is described below.

The LMS Algorithm

The Steepest Descent LMS algorithm, also known as the Widrow-Hoff algorithm, does not require explicit measurements of correlation functions, nor does it involve matrix inversion (Ref. 10:1144). It is based on a recursive algorithm in which the weight vector is changed at each iterative step along the direction of the negative gradient of the mean square error (Ref. 21:213).

An explanation of the steepest descent LMS algorithm is as follows. An initial estimate is made of the optimal weight vector Wopt(j); this first estimate will probably be just a guess, say W(0). Then, the cost function value is computed by evaluating $E[e^2(j)]$ at W = W(0) (Ref. 42:18).

Next, the gradient of the cost function is found by differentiating $E[e^2(j)]$ with respect to W. When the gradient is evaluated at W = W(0), it indicates the direction and rate of increase of $E[e^2(j)]$. Note that the direction of the gradient is the direction of the maximum rate of change (Ref. 42:19).

Now, update W in such a way that the evaluated cost function $E[e^2(0)]$ is reduced by the largest amount possible. Thus, the next weight vector, W(1), becomes

$$W(1) = W(0) - Ks\nabla_w^T\{E[e^2(0)]\}_{W=W(0)} \qquad (42)$$

where Ks is a constant which controls the rate at which W(j) moves toward Wopt.

For the general case, ie, at any iteration step, say j+1, the algorithm is given by

$$W(j+1) = W(j) - Ks\nabla_w^T\{E[e^2(j)]\}_{W=W(j)} \tag{43}$$

However, in actuality $E[e^2(j)]$ is not available and must be approximated by $e^2(j)$. Therefore, the algorithm becomes

$$W(j+1) = W(j) - Ks\nabla_w^T [e^2(j)]_{W=W(j)} \tag{44}$$

Observe that

$$\nabla_w[e^2(j)] = 2e(j)\nabla_w[e(j)] \tag{45}$$

and that

$$\nabla_w[e(j)] = \nabla_w[d(j) - W^T(j)X(j)] = -X^T(j) \tag{46}$$

Therefore

$$\nabla_w[e^2(j)] = -2e(j)X(j) \tag{47}$$

The final form of the algorithm is obtained after substituting equation (47) into equation (44):

$$W(j+1) = W(j) + 2Kse(j)X(j) \tag{48}$$

With the proper choice of Ks, the value of the weight vector, W, converges to the optimal weight vector, Wopt, as the number of iterations, j, increases without limit.

Adaptive Transversal Filter

Recall that the input to the adaptive linear combiner is a set of n sampled measurements at some time, t. Now let each sampled measurement, Xi(t), be a different delayed version of the same signal X1(t) such that

$$X2(t) = X1(t-\Delta)$$

$$X3(t) = X2(t-\Delta) = X1(t-2\Delta)$$

$$\vdots$$

$$Xn(t) = X1(t-(n-1)\Delta)$$

where Δ is some delay.

The above desrcibed scheme, as illustrated by Figure III-2, results in a so-called adaptive transversal filter. This is in fact the filter type that will be of concern in the work that follows.

Observe that equation (28) can now be written as

$$y(t) = W1X1(t) + W2X1(t-\Delta) + \ldots + WnX1(t-(n-1)\Delta) \quad (49)$$

Frequently, in the literature, the first weight coefficient, W1, is called Wo. Adopting this practice and dropping the subscript on X1, equation (49) becomes

$$y(t) = WoX(t) + W1X(t-\Delta) + \ldots + Wn-1X(t-(n-1)\Delta) \quad (50)$$

Now let m=n-1, so that y(t) becomes

$$y(t) = WoX(t) + W1X(t-\Delta) + \ldots + WmX(t-m\Delta) \quad (51)$$

Since x(t) is sampled with a period equal to Δ, y(t) can be

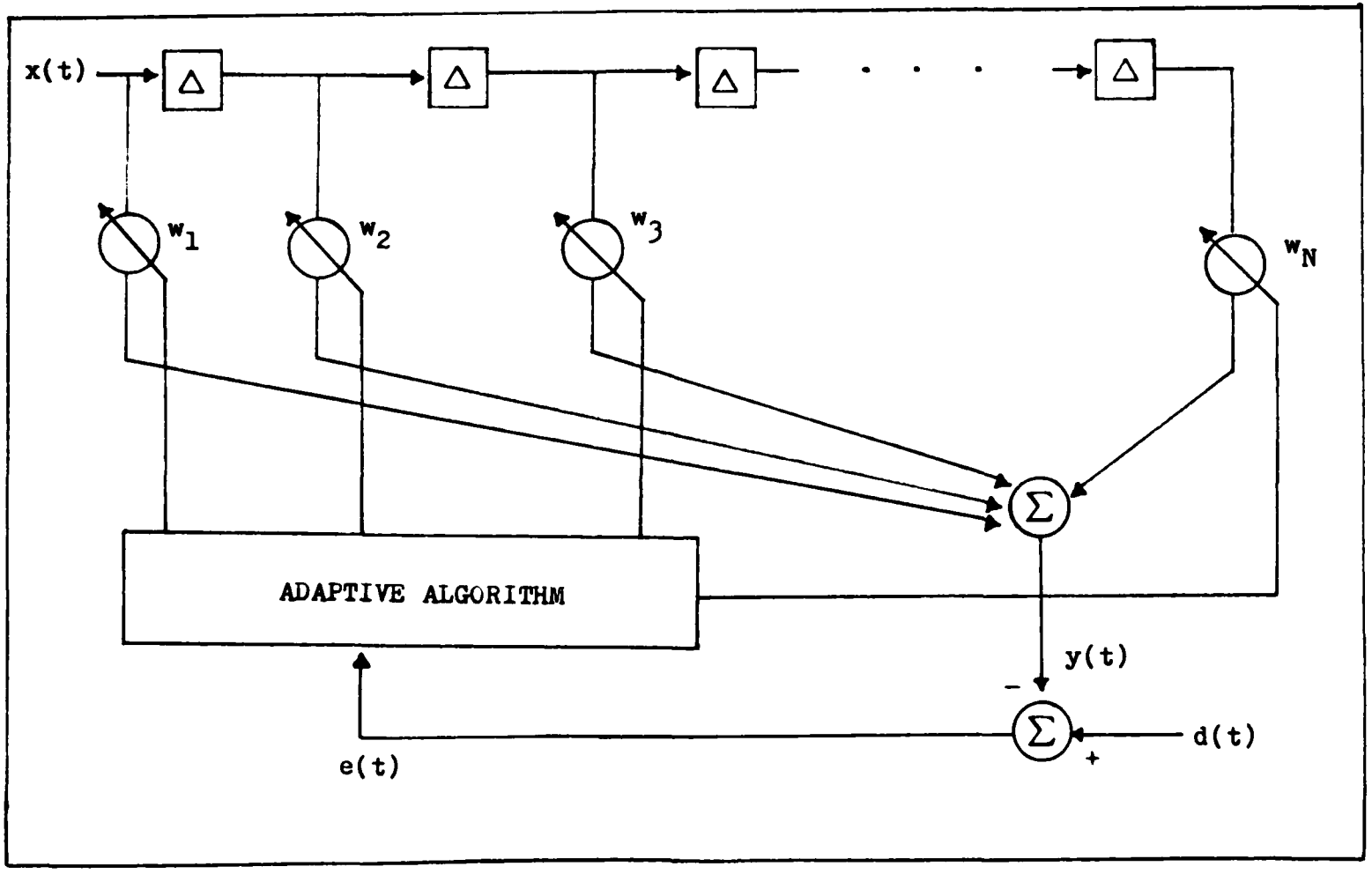


Figure III-2 Adaptive Transversal Filter

expressed as a discrete-time signal (Refs. 42:11; 43:3). Therefore, y(t) becomes

$$y(k) = WoX(k) + W1X(k-1) + \ldots + WmX(k-m) \tag{52}$$

Using the z transform (Ref. 43:34), y(k) becomes

$$\begin{aligned} Y(z) &= WoX(z) + W1z^{-1}X(z) + \ldots + Wmz^{-m}X(z) \\ &= (Wo + W1z^{-1} + \ldots + Wmz^{-m})\, X(z) \end{aligned} \tag{53}$$

The transfer function of the filter, H(z), is given by Y(z)/X(z). Therefore,

$$H(z) = Wo + W1z^{-1} + \ldots + Wmz^{-m} \tag{54}$$

The Soft-Constraint LMS Algorithm

Observe from Figure III-2 that in the conventional LMS algorithm the y(t) is some filtered version of x(t) and that the desired response, d(t), is required to obtain e(t). However, the desired response is unknown, so let d(t)=0. Since y(t) will be as close to d(t) as possible, y(t) will approach zero instead of x(t).

Therefore, a new algorithm is needed that can utilize d(t)=0. The desired algorithm should (1) minimize the power in e(t), that is, $E[e^2(t)]$ is minimized, (2) produce a desired gain in a passband determined by some selected frequencies, and (3) produce zero gain elsewhere.

The goal of the soft-constraint algorithm is to minimize $E[e^2(t)]$ subject to the "soft" constraints on the frequency response of the filter. "The constraints are called soft

because, unlike constraints in most optimization problems, they can be violated (not satisfied exactly)" (Ref. 20:10). The constraints are specified at c frequencies, W1, W2, ... , Wc and are given by

$$\begin{aligned} H(jW1) &= A1*\exp[j\angle H(jW1)] \\ H(jW2) &= A2*\exp[j\angle H(jW2)] \\ &\vdots \\ H(jWc) &= Ac*\exp[j\angle H(jWc)] \end{aligned} \tag{55}$$

where the magnitude of H(jW1) = A1 and the phase of H(jW1) = $\angle$H(jW1). Figure III-3 further illustrates the above relationships.

Friedlander and Morf (Ref. 19:381) claim that FIR linear-phase filters are "important for applications where frequency dispersion due to nonlinear phase is harmful, e.g., speech processing and data transmision." They also claim that "relatively little work seems to have been done on adaptive linear-phase filters." Therefore, to insure a linear-phase filter, the constraint frequencies were chosen to be evenly spaced within the frequency band of interest in the computer simulation.

To derive the soft-constraint LMS algorithm, a set of matrices A, B, and H, called respectively, the constraint matrix, the weighting matrix, and the frequency matrix, are defined as follows. The constraint matrix, A, is a function of A1, A2, ... , Ac and the required phases associated with frequencies W1, W2, ... , Wc. The weighting matrix, B, is a

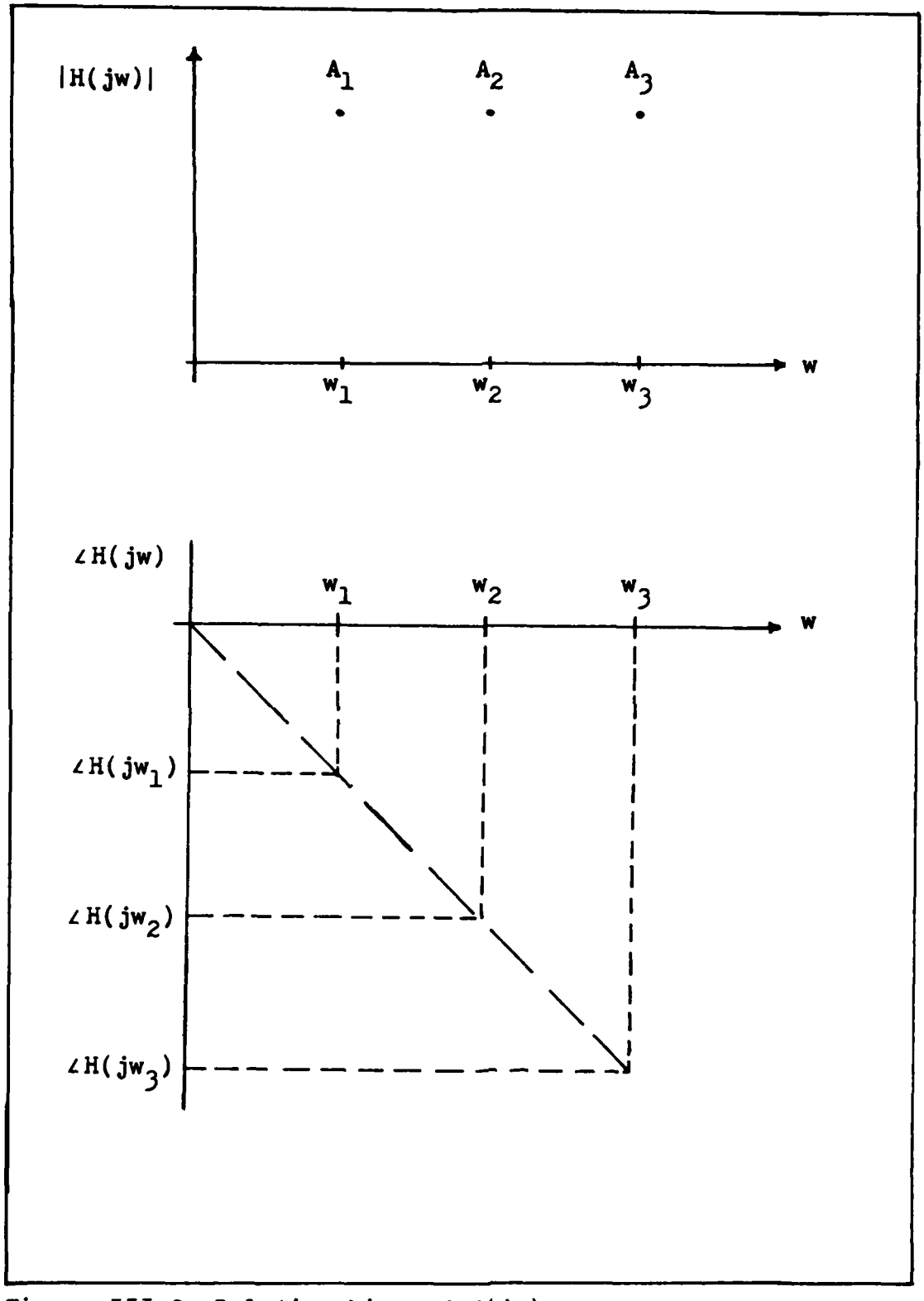


Figure III-3 Relationships of H(jw)

function of the weight assigned to each constraint frequency. The frequency matrix, H, is a function of the frequencies at which constraints are specified.

To implement the algorithm in a computer model the A matrix, which is complex, is resolved into its real and imaginary parts. Thus, for an adaptive filter with n+1 weights at a particular constraint frequency, Wi, the ith submatrix of the matrix A, the Ai matrix, would be given by

$$Ai = \begin{bmatrix} 1 & \cos(-WiT) & \cos(-2WiT) & \dots & \cos(-nWiT) \\ 0 & \sin(-WiT) & \sin(-2WiT) & \dots & \sin(-nWiT) \end{bmatrix}$$

The above matrix is "stacked" with the other Ak matrices associated with each constraint frequency Wk to obtain the final A matrix. Thus, for C constraint frequencies, the A matrix is given by

$$A = \begin{bmatrix} 1 & \cos(-W1T) & \cos(-2W1T) & \dots & \cos(-nW1T) \\ 0 & \sin(-W1T) & \cos(-2W1T) & \dots & \cos(-nW1T) \\ & \vdots & \vdots & & \\ 1 & \cos(-WcT) & \cos(-2WcT) & \dots & \cos(-nWcT) \\ 0 & \sin(-WcT) & \sin(-2WcT) & \dots & \sin(-nWcT) \end{bmatrix}$$

Since the A matrix was expanded into real and imaginary parts, the number of terms in the H vector must be doubled. Thus, the H vector becomes

$$H = \begin{bmatrix} \cos(-W1) \\ \sin(-W1) \\ \cos(-W2) \\ \sin(-W2) \\ \vdots \\ \cos(-Wc) \\ \sin(-Wc) \end{bmatrix} \qquad (56)$$

The requirement for the algorithm is to minimize $E[e^2(t)]$ subject to the constraint $(AW-H)^T B(AW-H)$. One can show that minimizing

$$E[e^2(t)] + (AW-H)^T B(AW-H)$$

using the steepest descent philosophy gives (Ref. 20)

$$W(j+1) = W(j) + 2Kse(j)X(j) - 2KsA^T B[\ AW(j)-H\ ] \qquad (57)$$

or, equivently,

$$W(j+1) = (I - 2KsA^T BA)W(j) + 2Kse(j)X(j) + 2KsA^T BH \qquad (58)$$

where

A is the constraint matrix

B is the weighting matrix

H is the frequency vector

I is the identity matrix

Observe that once the constraint frequencies are chosen the terms of the A matrix and H vector become constant and do not change with each iteration of the algorithm. The values of the diagonal of the B matrix are user selectable at the

beginning of the computer simulation. These values of the constraint weighting matrix determine the relative importance of meeting the constraint at each constraint frequency. Note that by setting B = 0 in equation (57) the LMS algorithm given by equation (48) is obtained. Thus, it can be seen that the LMS algorithm is a special case of the soft-constraint LMS algorithm.

Theoretically, the adaptive filter's frequency response shown in Figure III-4a is possible with the proper choice of constraint weights. However, it was felt that more effective notches could be obtained by not requiring that the adaptive filter also act as a low pass filter (Figure III-4b). Therefore, a tenth order Butterworth low pass filter (Figure III-4c) was used in tandem with the adaptive filter in the computer model.

The equivalent algorithm given by equation (58) is implemented by the computer model discussed in the next chapter.

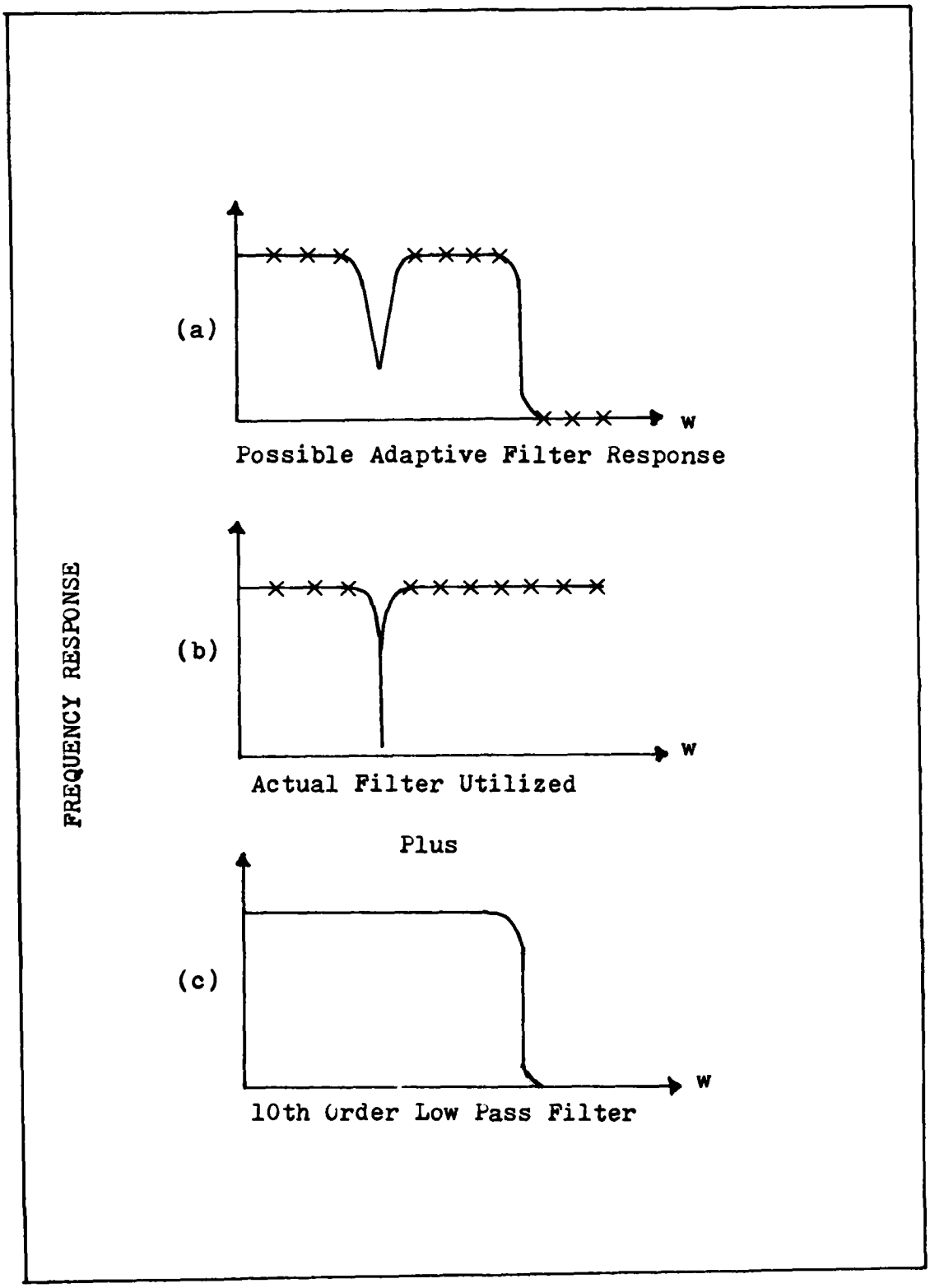


Figure III-4 Frequency Responses of Filters

IV. COMPUTER MODEL

Simulation Method Overview

To fully comprehend the results of this thesis effort, an understanding of the method of simulation must first be developed. The computer program presented in Appendix A models a spread spectrum receiver preceded by an adaptive notch filter using the soft-constraint algorithm. Figure IV-1 shows a simplified block diagram of the receiver and filter computer model. The program generates the spread spectrum signal, S(K), the jammer's signal, J(K), and the noise, N(K), in the digital domain and adds them together to form the received signal, X(K). This received signal is viewed as sampled data points at instances K, K+1, K+2, ... and enters the adaptive filter. At each of these instances, the weight coefficients of the adaptive filter are computed and used to obtain the filter output, Y(K). After a user selected number of iterations, the final weight coefficients are used to compute and plot the filter's frequency response. Next, the same SS, jammer, and noise signals that had contributed to the received signal are separately sent through the final weighted filter. Then, based upon the filter output, their separate powers are computed. These signal powers are then presented to the SS receiver and the receiver's output powers are calculated along with various signal to noise and signal to jammer ratios.

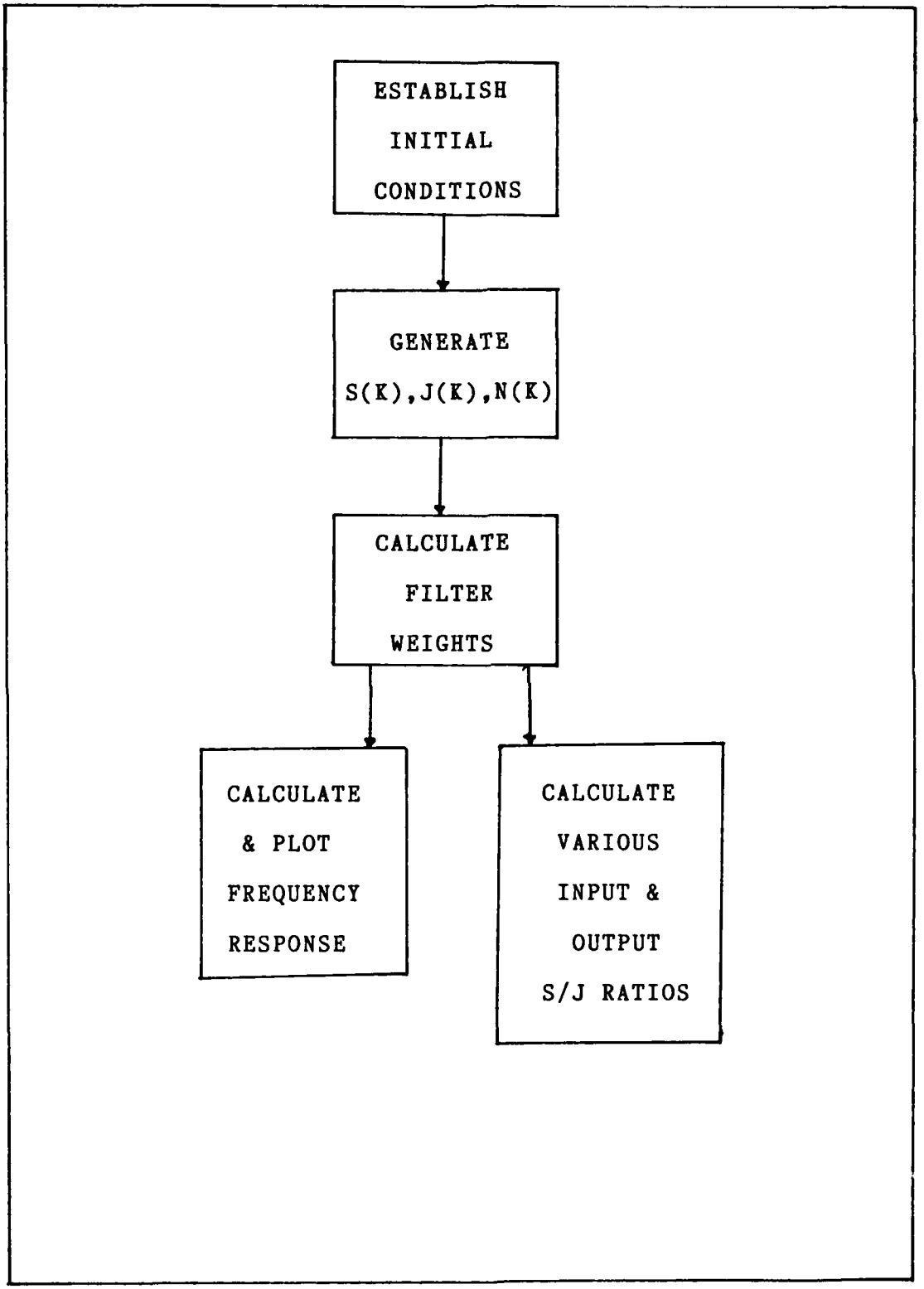


Figure IV-1 Simplified Flow Diagram

Error Prevention

In order to prevent the truncation and roundoff errors that are characteristic of digital computers, two preventive measures were taken. First, the method of frequency scaling is utilized so that all frequency values are much smaller than they would be in actual systems. For example, in the computer simulation a PG of 100 is a nominal figure, whereas, in actual systems PG can range from 10^3 to 10^6. The second preventive measure is to simulate the receiver in the baseband or low pass region. This is the same as considering perfect carrier recovery as noted in the assumptions in the introductory chapter.

In addition, to simplify the calculations and computer model, the power of the received spread spectrum signal is taken to be 1.0 and the signal pulse duration is taken to be 1.0 seconds. Recall, that for a pulse of unit width, most of the "information" or "area" in the frequency domain lies between -2Π and $+2\Pi$. Therefore, the receiver's input filter has a bandwidth from -2Π to $+2\Pi$. This baseband bandwidth and the fact that the signal power equals one must be kept in mind in order to select reasonable values for the AWG noise power and the jammer's power and frequency.

Preliminary Steps

Before the main program, ADAPSS, can be executed a file of random numbers with a normal (0,1) distribution must be available. To obtain the random numbers the subroutine "GGNML" of the International Mathematical and Statistical

Libraries (IMSL) was used in the following program:

```
PROGRAM NEWRN
INTEGER NR
REAL R(1000)
DOUBLE PRECISION DSEED
OPEN(16,FILE='RNFILE')
REWIND 16
NR=1000
DSEED=123457D.0
CALL GGNML(DSEED,NR,R)
WRITE(16,'(F12.8)')R
END
```

Once the generation of the random numbers is completed the ADAPSS program can be started. Of course, the first steps in the program declare the variables and assign initial conditions. Then, the program conputes the constant values of the A matrix and H vector based upon the constraint frequencies as explained in Chapter III.

The program then asks for the user selected input parameters such as the number of hops made by the jammer, the jammer's power and frequency at each hop, the noise power, the processing gain of the SS receiver, and the convergent factor of the soft-constraint LMS algorithm. The user also selects the B matrix weighting value associated with each constraint frequency.

Finally, the user selects the number of iterations that the adaptive filter algorithm is to perform before arriving at a final set of weight coefficients. Put another way, the user selects the number of data points or inputs into the adaptive filter desired. In either case, the number selected is added to 19 before input to the computer program. This is

necessary because the adaptive filter is modeled as having 19 delay elements and therefore, the first 19 data points or samples are assigned as initial conditions.

After all the user variables have been selected, the program computes the values of several matrices that depend on the B matrix weighting values. Then the program starts the "do loop" which is the basis of the iterative process to solve the adaptive filter's algorithm. The first steps inside the do loop are used to calculate the jammer, noise, and spread spectrum signals.

Signal, Noise, and Jammer Calculation

At the beginning of each iteration of the do loop a new random number is read from the random number file. The random number is used in the calculation of the signal and noise.

If the random number is positive, then the value of the SS signal is +1 and if the random number is negative, then the value of the SS signal is -1. Since the sample time, T, is 0.2 seconds and the duration of the SS signal is 1.0 second, the sign of the random number is checked every 5th iteration to determine the value of the SS signal for the next 5 data points.

The value of the noise is determined by passing the normally distributed random number through a low pass 2nd order Butterworth spectral shaping filter. The amplitude of the filter, AMP, depends upon the user selected noise power, NPWR, and was found experimentally to be given by

$$AMP = \sqrt{NPWR/0.17} \tag{59}$$

It turns out that the value of the noise is calculated from the current random number as well as the previous random number and noise value.

The jamming signal is given by

$$J(K) = \sqrt{2*JPWR}\ \cos(FJ*K*T) \tag{60}$$

where

JPWR = power of the jammer

FJ = frequency of the jammer

K = iteration of the do loop

T = sampling period

The SS signal, S(K), the noise, N(K), and the jamming signal, J(K), are added together to obtain the data points, X(K), applied to the adaptive filter input.

Adaptive Filter Weights

Recall from Chapter III that the soft-constraint LMS algorithm uses the data points, x(j), and the error signal, e(j), to determine the filter weights. Recall also that

$$e(j) = d(j) - y(j) \tag{35}$$

However, since the desired signal, d(j), is unknown, let d(j) = 0, and thus, e(j) = -y(j). The next step in the do loop calculates y(j) where j has been replaced by K in the program. It can be seen from Figure III-2 that

$$Y(K) = X(K)W1 + X(K-1)W2 + \ldots + X(K-19)W20. \quad (61)$$

Thus, using e(j) = -Y(K), X(K), and the matrices calculated earlier, the filter weights are computed by using the algorithm given by equation (58) just before the end of the do loop. The steps in the do loop are repeated as often as desired to arrive at the final filter weights by the iterative process. The final weights are passed to two subroutines, XFER and PWROUT, before the end of the main program.

Subroutines

The XFER subroutine uses the final filter weights to find the transfer function of the adaptive filter. The magnitude of the transfer function is found at increments of 0.04 radians. The transfer function is multiplied by a 10th order Butterworth low pass filter to simulate the input bandpass filter of an actual receiver. The results at each frequency are stored in different files associated with each jammer's hop for subsequent plotting.

The PWROUT subroutine computes the signal, noise, and jammer powers at the adaptive filter output. These powers are found by passing 200 sample points of each signal through the filter with the weights set to the final values determined by the main program. The output of the filter, Y(K), for each type of signal, is squared and the sum of the squares are divided by 200 to find the average power. The various signal to noise and jammer ratios are multiplied by

the processing gain, PG, to find the ratios associated with the output of the spread spectrum receiver.

There are several other subroutines used to print out the values of the various matrices calculated during the program. While the values of these matrices can be interesting, these subroutines were generally used for debugging the program.

V. RESULTS

The results of this thesis effort are presented mainly in the form of plots in this chapter. As a starting point, the power spectral density (PSD) of the input (received) SS signal and noise are shown in Figures V-1 and V-2, respectively. Although the frequency axis only extends to 8 rad/sec, it is easy to see that most of the SS signal's energy is below 2Π rad/sec as expected for one second duration pulses. The PSD of the input (received) jammer signal is not plotted since the PSD of a CW sinusoid is merely an impulse.

The Constant Conditions

Although the computer program was written to prompt the user to select many parameters, some of the initial parameters are set in the beginning of the program. For example, the first nineteen data samples, X(K), and the initial twenty filter weights, are set to 0.05. Also, the ten constraint frequencies are set to 1.0, 2.1, 3.1, 4.2, 5.2, 6.3, 10, 12, 13, and 14 rad/sec. The sampling period, T, is passed into all the applicable subroutines as a variable, but is set to 0.2 seconds in the initial conditions.

In addition to the above conditions, all of the following results have in common some of the user selectable parameters. For example, the noise power was always selected

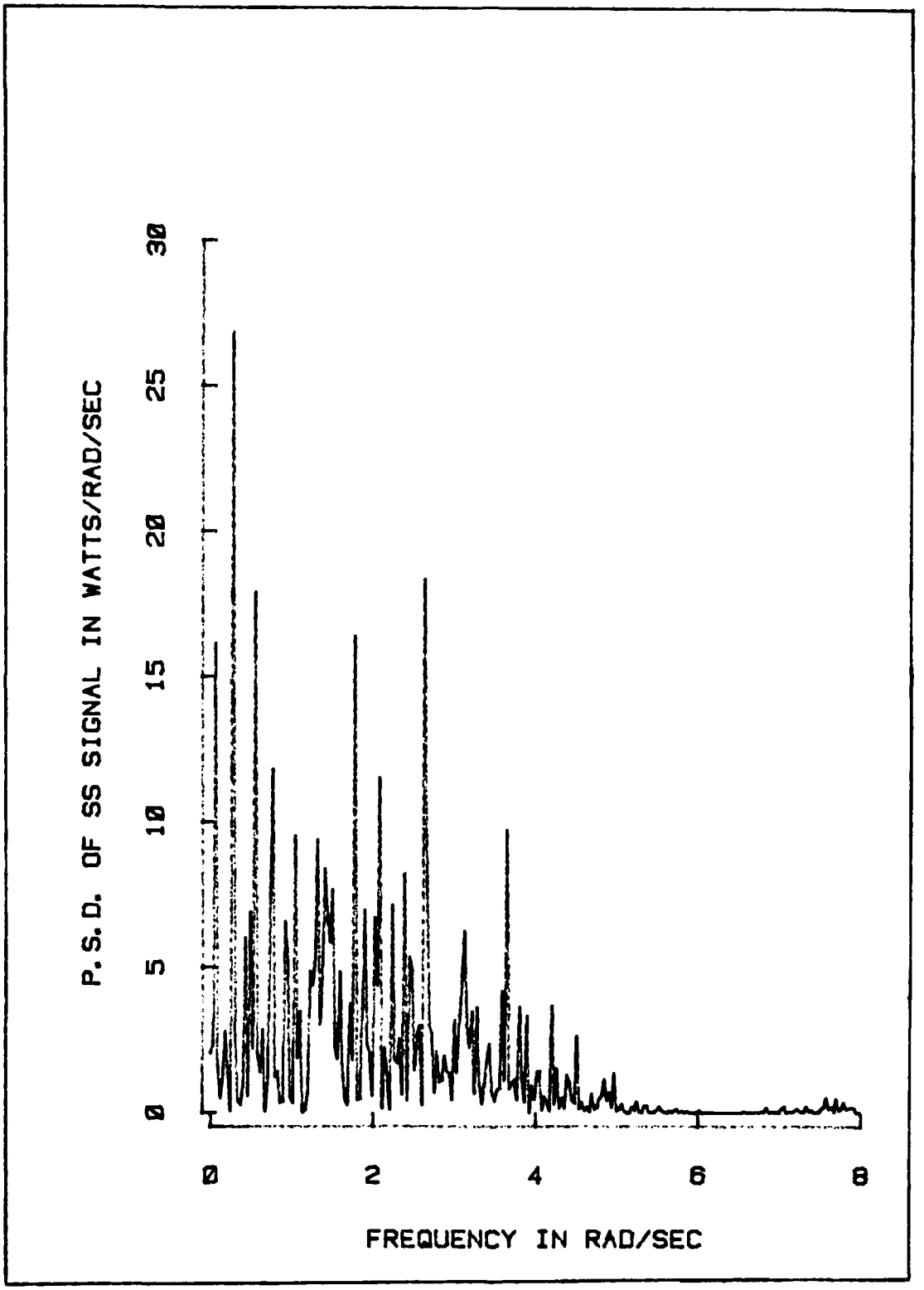


Figure V-1 Received SS Signal

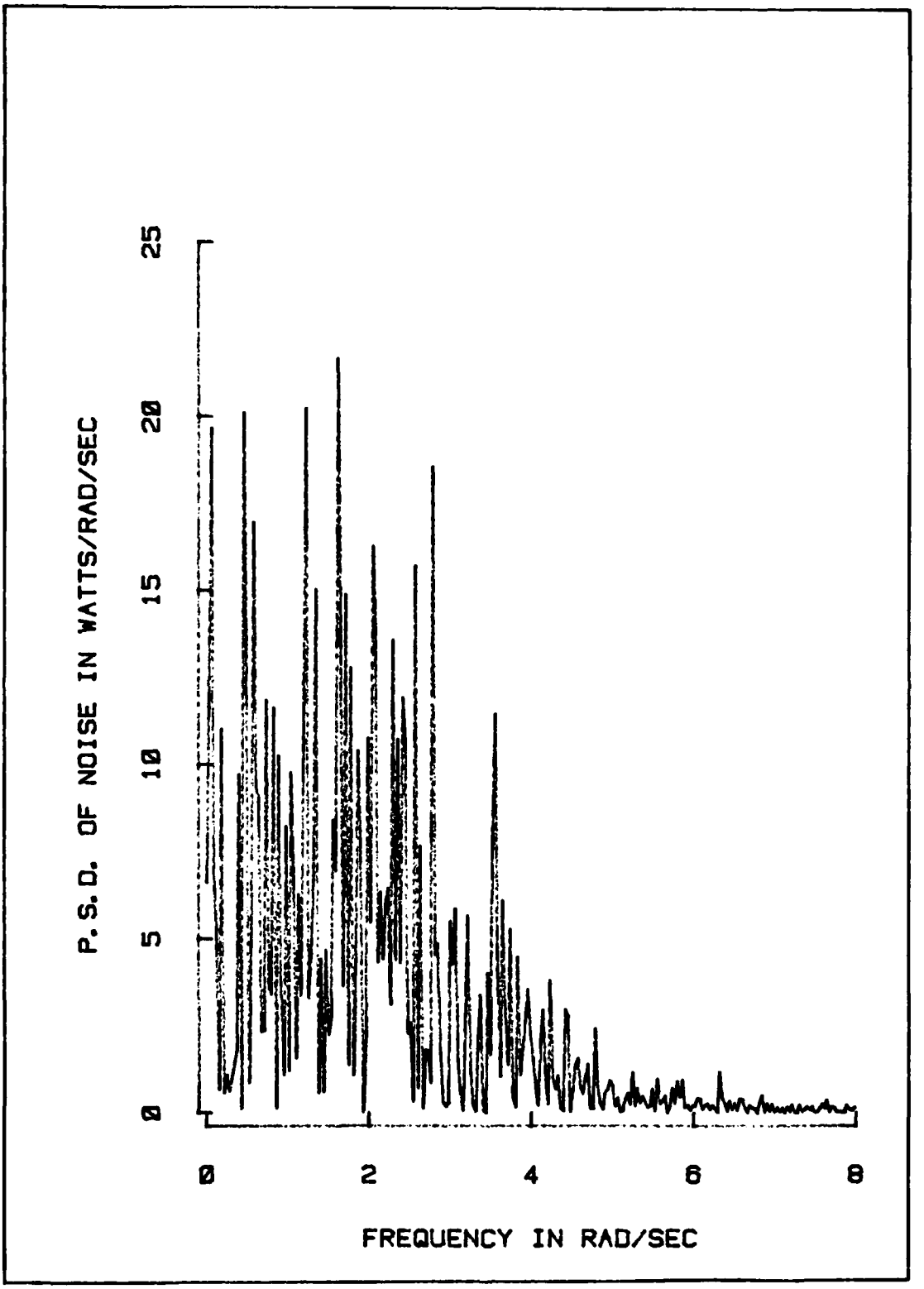


Figure V-2 Received Noise

as 0.8 and the convergent factor, Ks, was always selected to be -0.003. This value was found experimentally to give the best results. Although this value appears small, larger absolute values caused instability in the filter.

Finally, the transfer function of the adaptive filter before the reception of any signals is presented in Figure V-3. This baseline plot will help emphasize the changes made by the adaptive filter to reduce the effect of the jammer.

Single Frequency Jammer

This section shows how well the adaptive filter responds to a jamming signal located at a particular frequency. For all the results presented in this section the jammer's power was selected as 100 watts or 20 dB.

Figures V-4 to V-16 show the transfer function of the adaptive filter after a certain number of iterations of the soft-constraint LMS algorithm for a jammer located at 3.6 rad/sec. Table V-1 summarizes the results presented by these figures. Observe in Figure V-4 that after only 3 seconds (15 iterations times a sampling interval of 0.2 seconds) the notch has already begun to appear. This is even more interesting when one recalls that after 15 iterations the 19 element tapped delay line still contains 4 data points supplied as initial conditions by the computer program.

Observe also how the notch depth continues to vary even after a significant number of data samples have been analyzed. This is presumably due to the iterative nature of the algorithm. Nevertheless, all of the notches extend to at

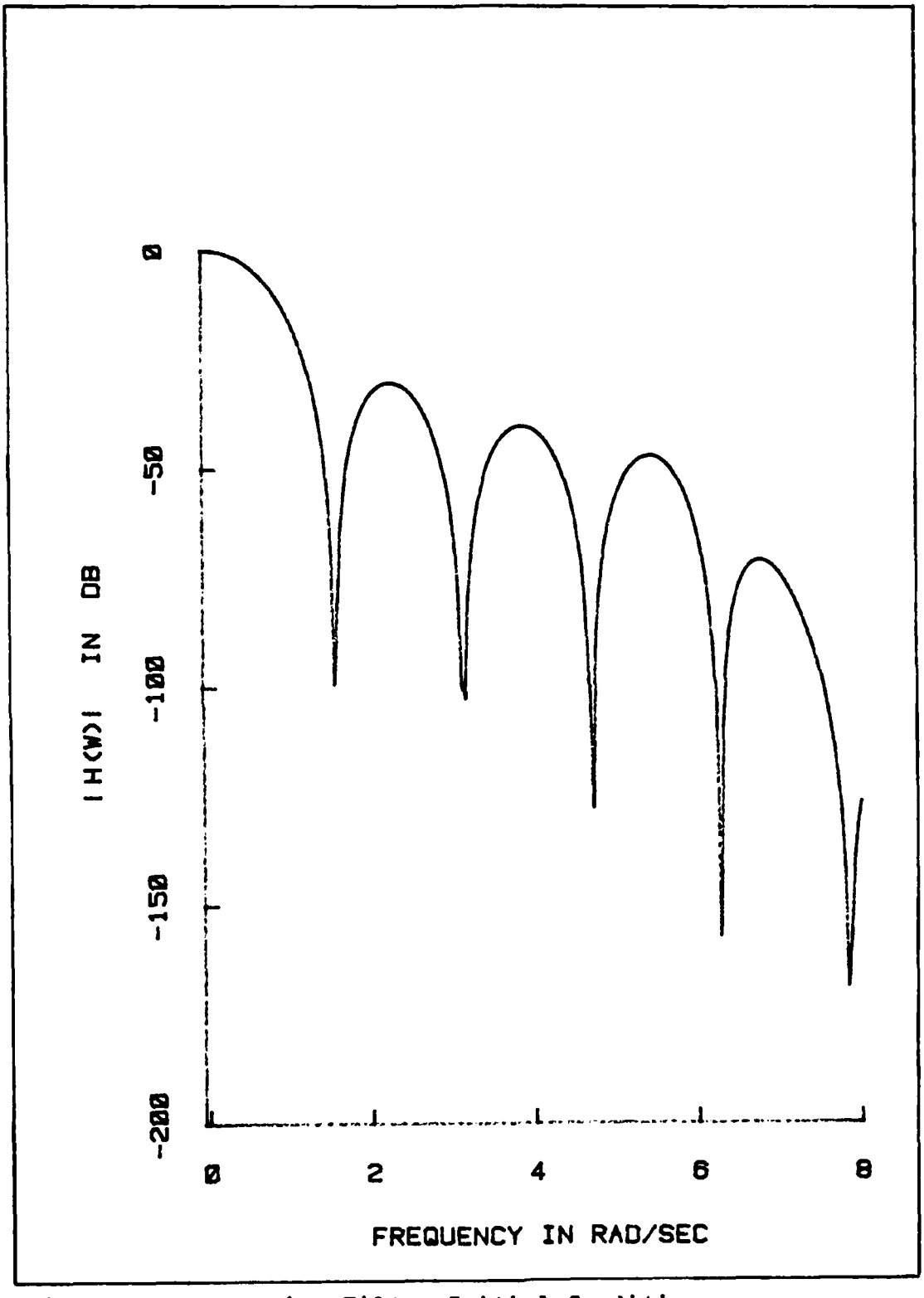


Figure V-3 Adaptive Filter Initial Condition

Table V-1

Summary of Notch Depths and Algorithm Iterations

For a 20 dB Jammer Located at 3.6 rad/sec

Figure	Iterations	Notch Depth (dB)
3	0	N/A
4	15	-36
5	24	-70
6	26	-34
7	35	-76
8	46	-52
9	49	-44
10	75	-37
11	100	-66
12	200	-35
13	500	-55
14	1000	-77
15	2000	-39
16	5000	-68

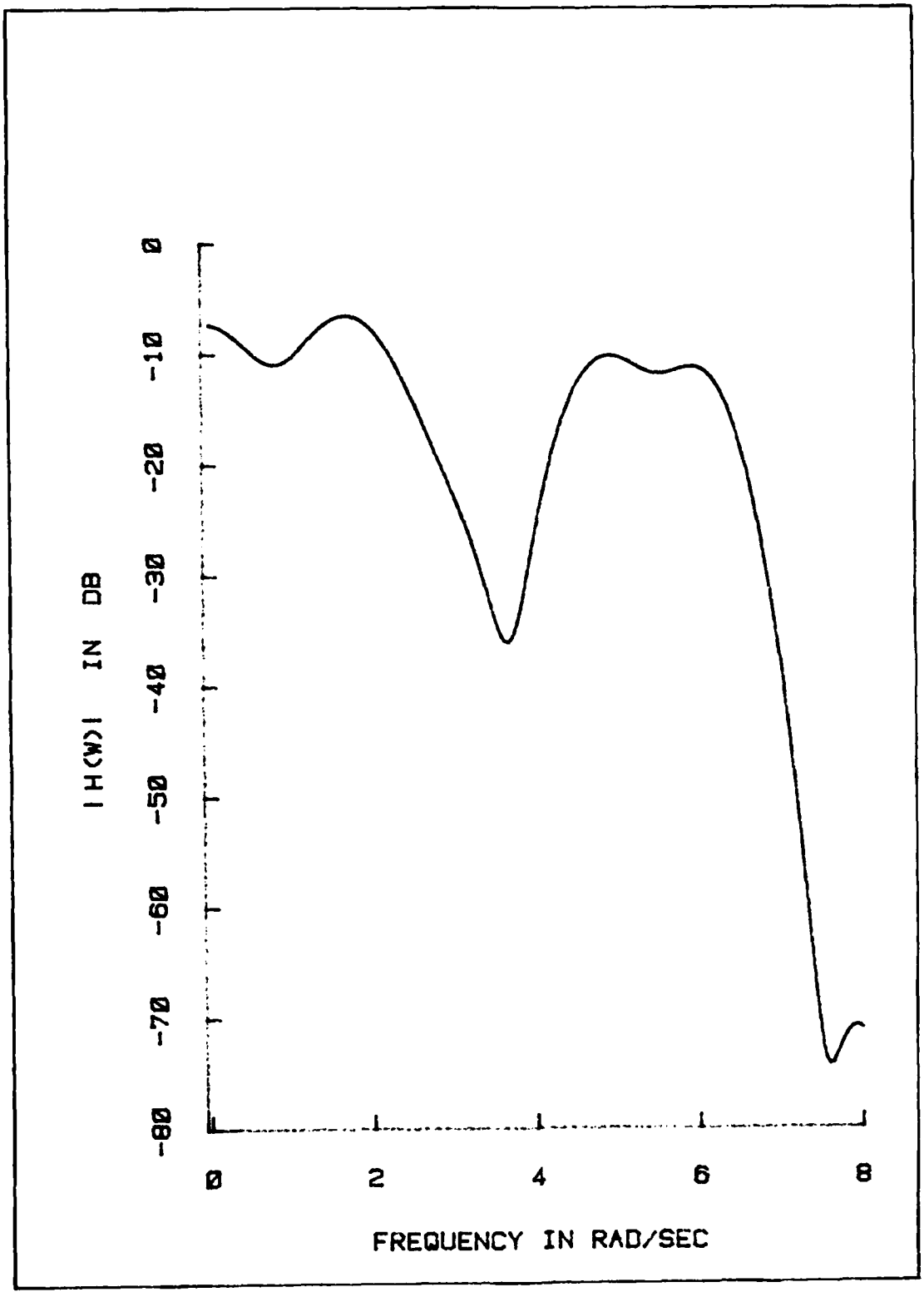


Figure V-4 15 Iterations

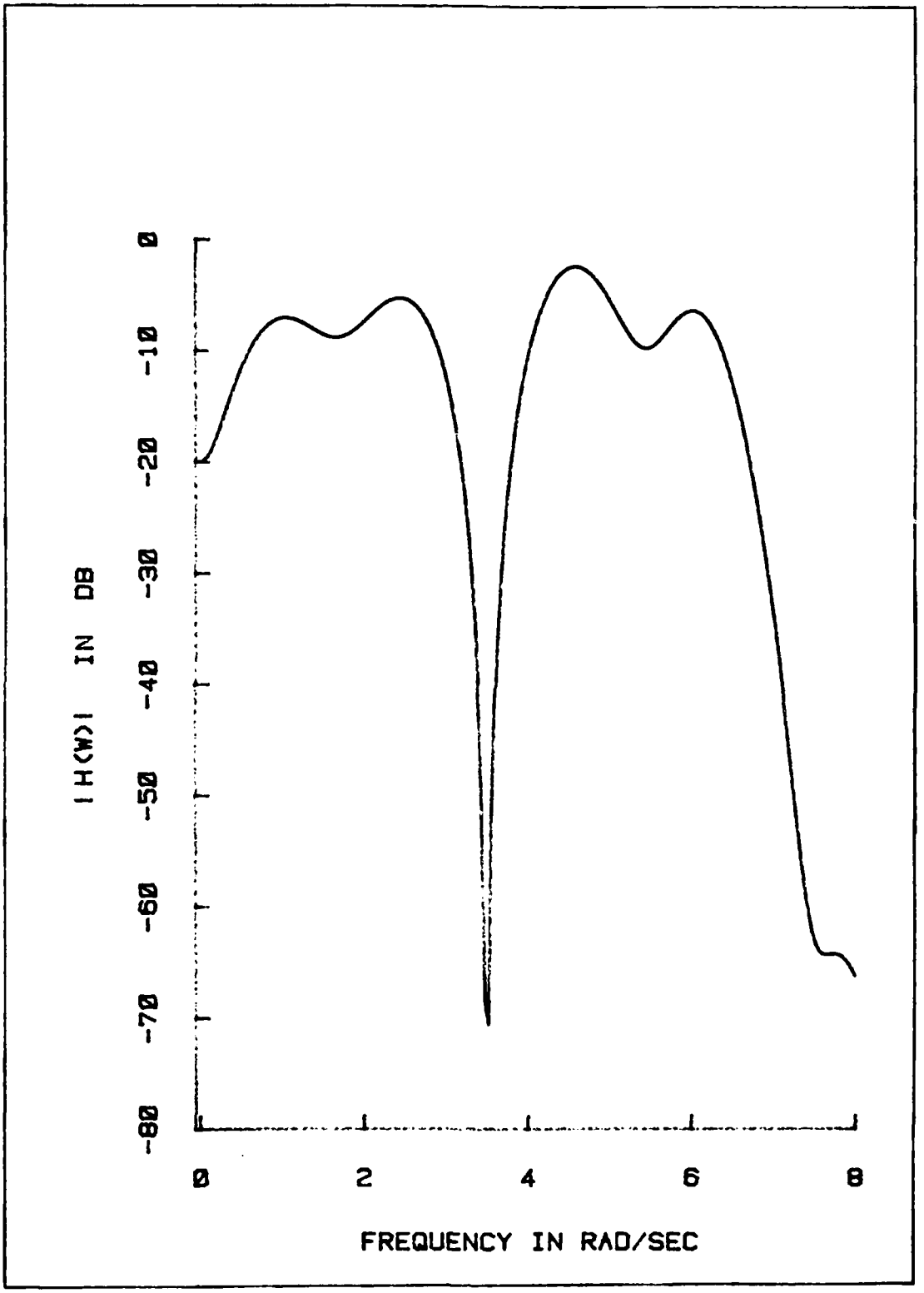


Figure V-5 24 Iterations

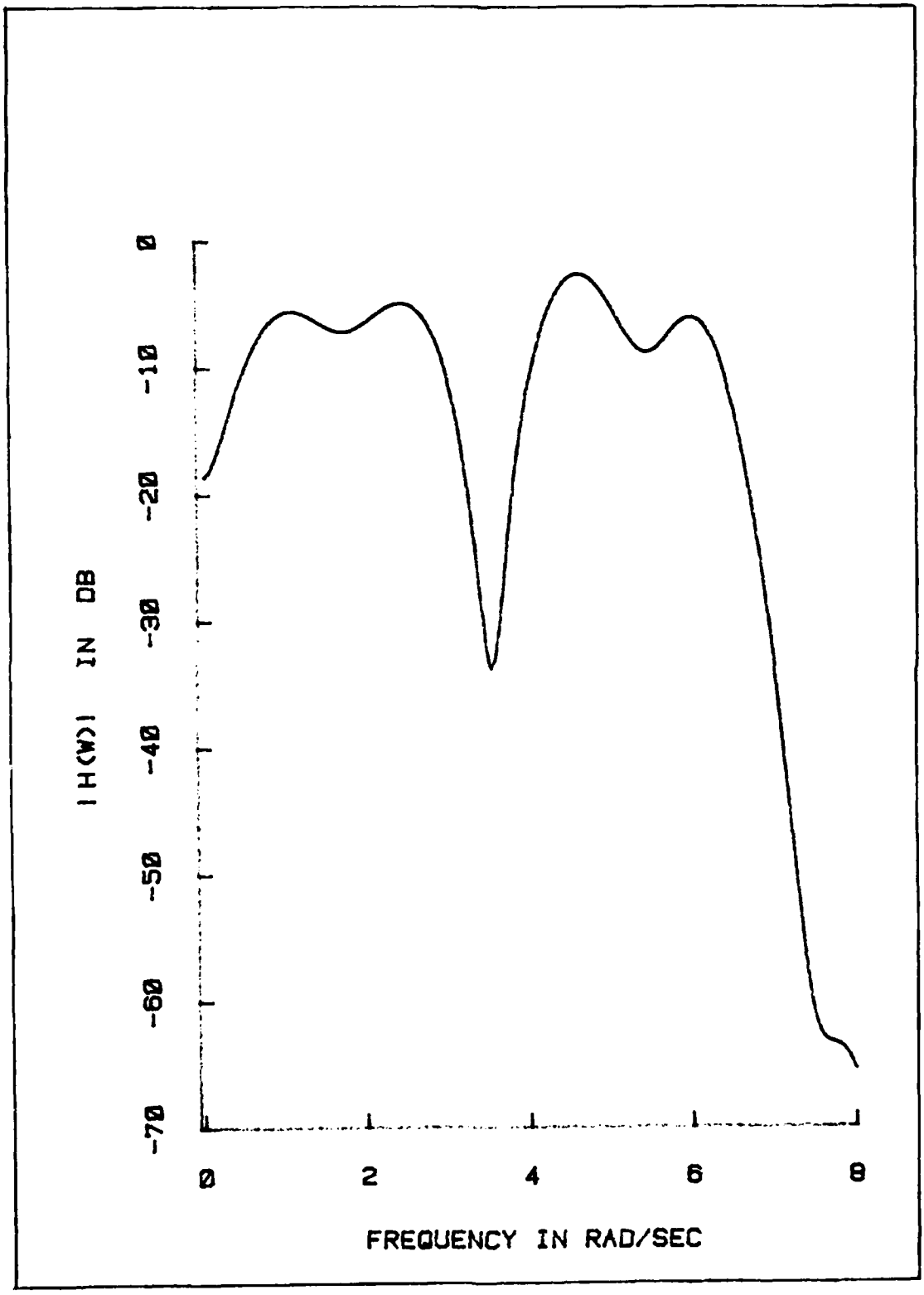


Figure V-6 26 Iterations

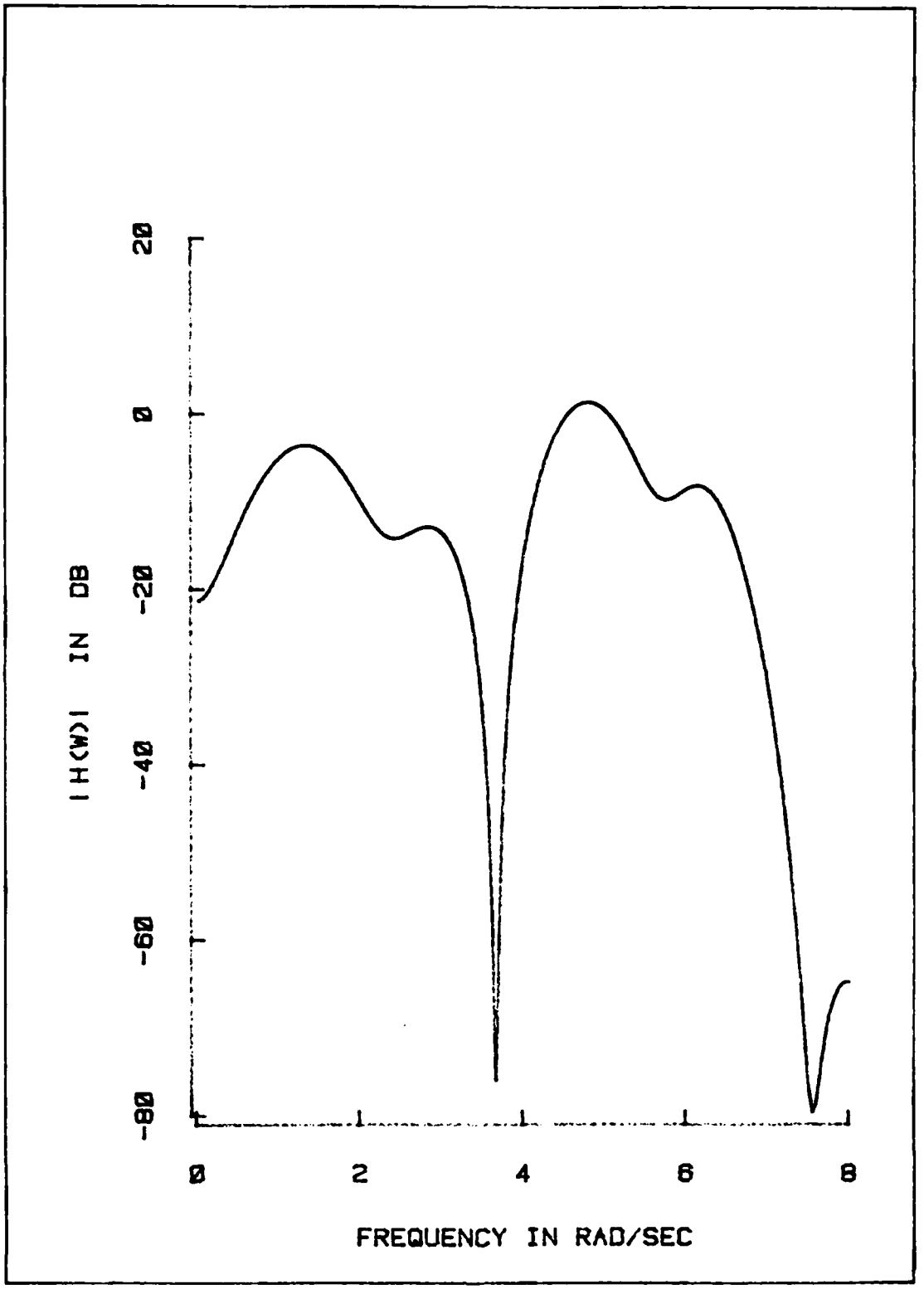


Figure V-7 35 Iterations

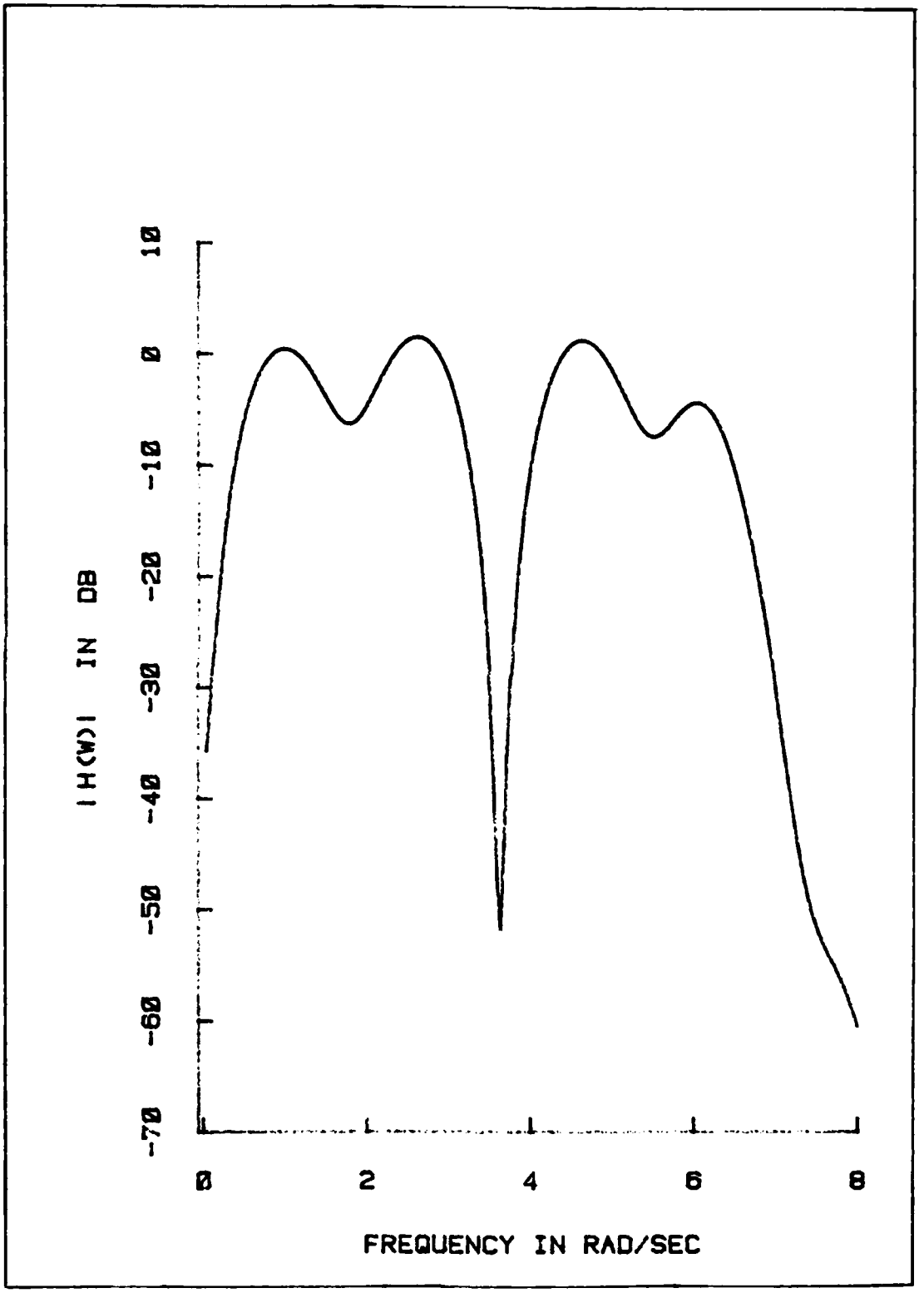


Figure V-8 46 Iterations

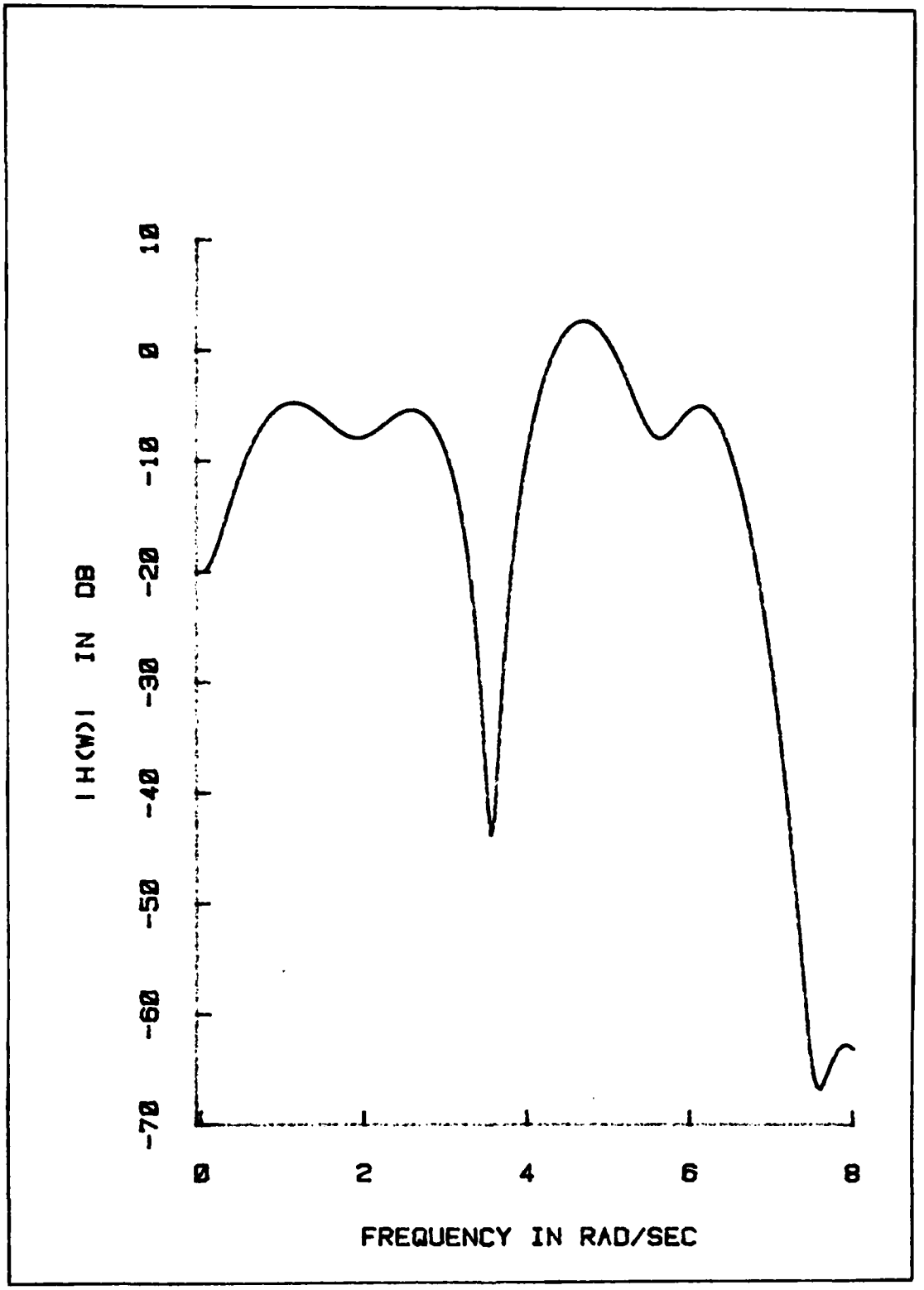


Figure V-9 49 Iterations

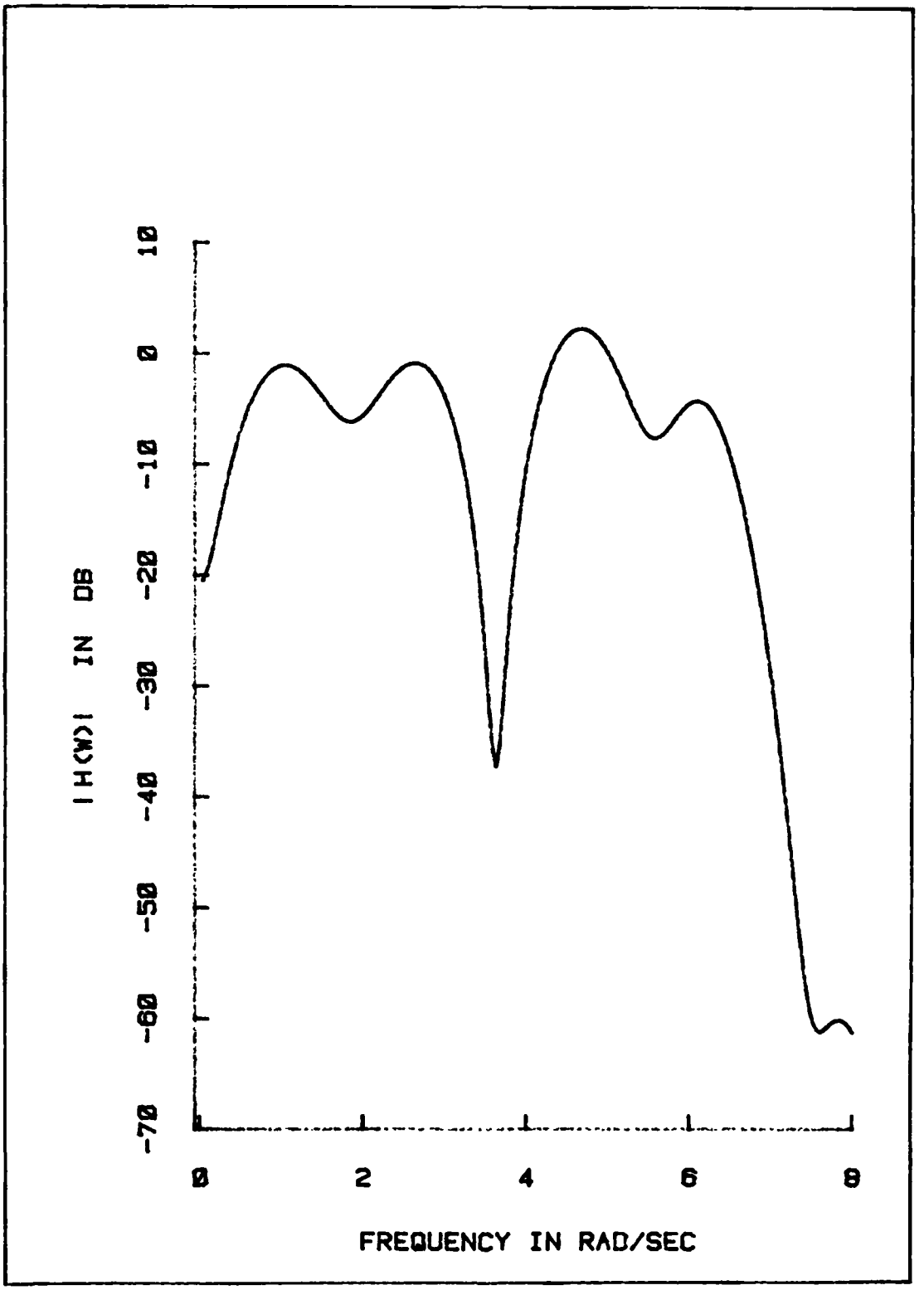


Figure V-10 75 Iterations

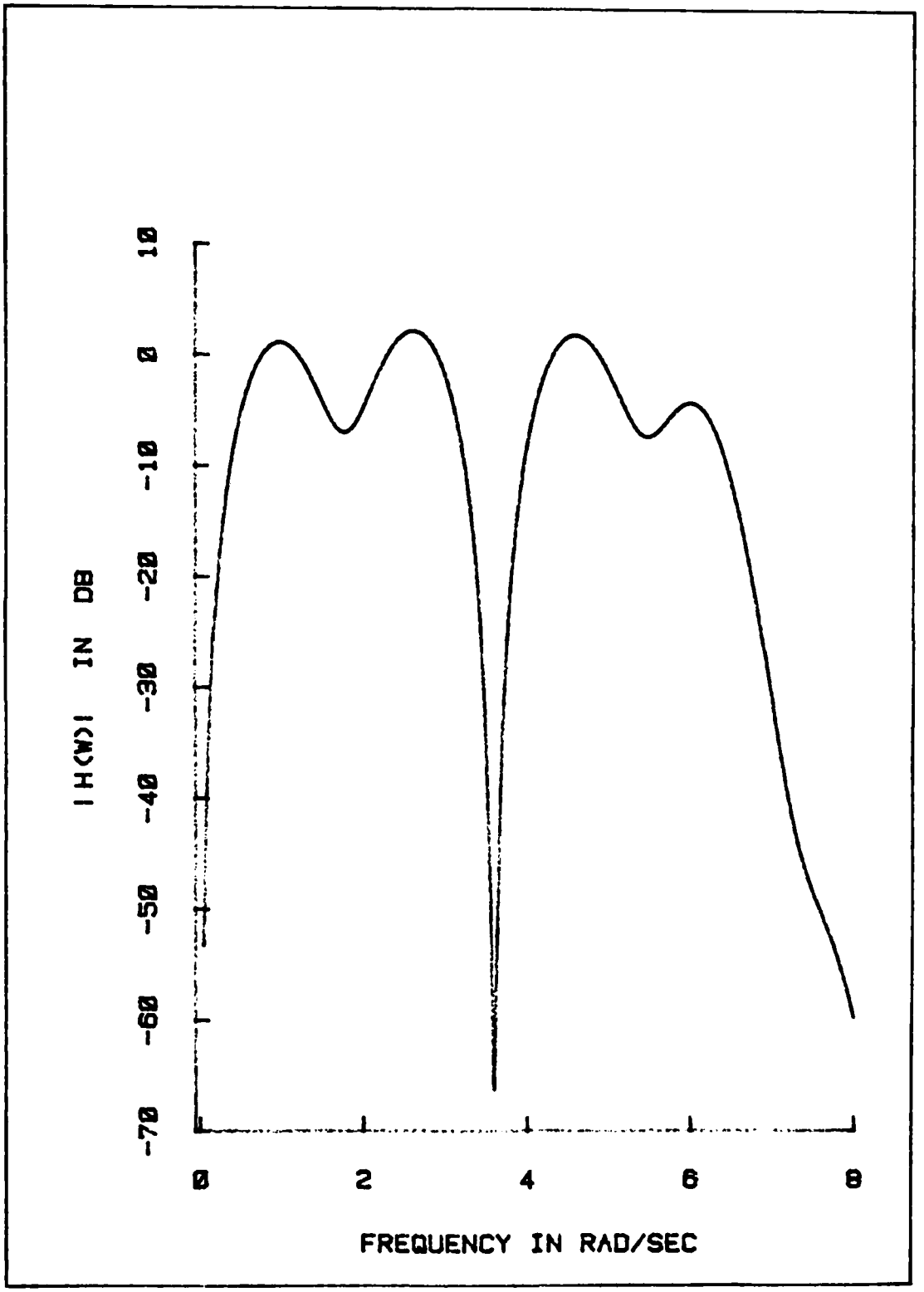


Figure V-11 100 Iterations

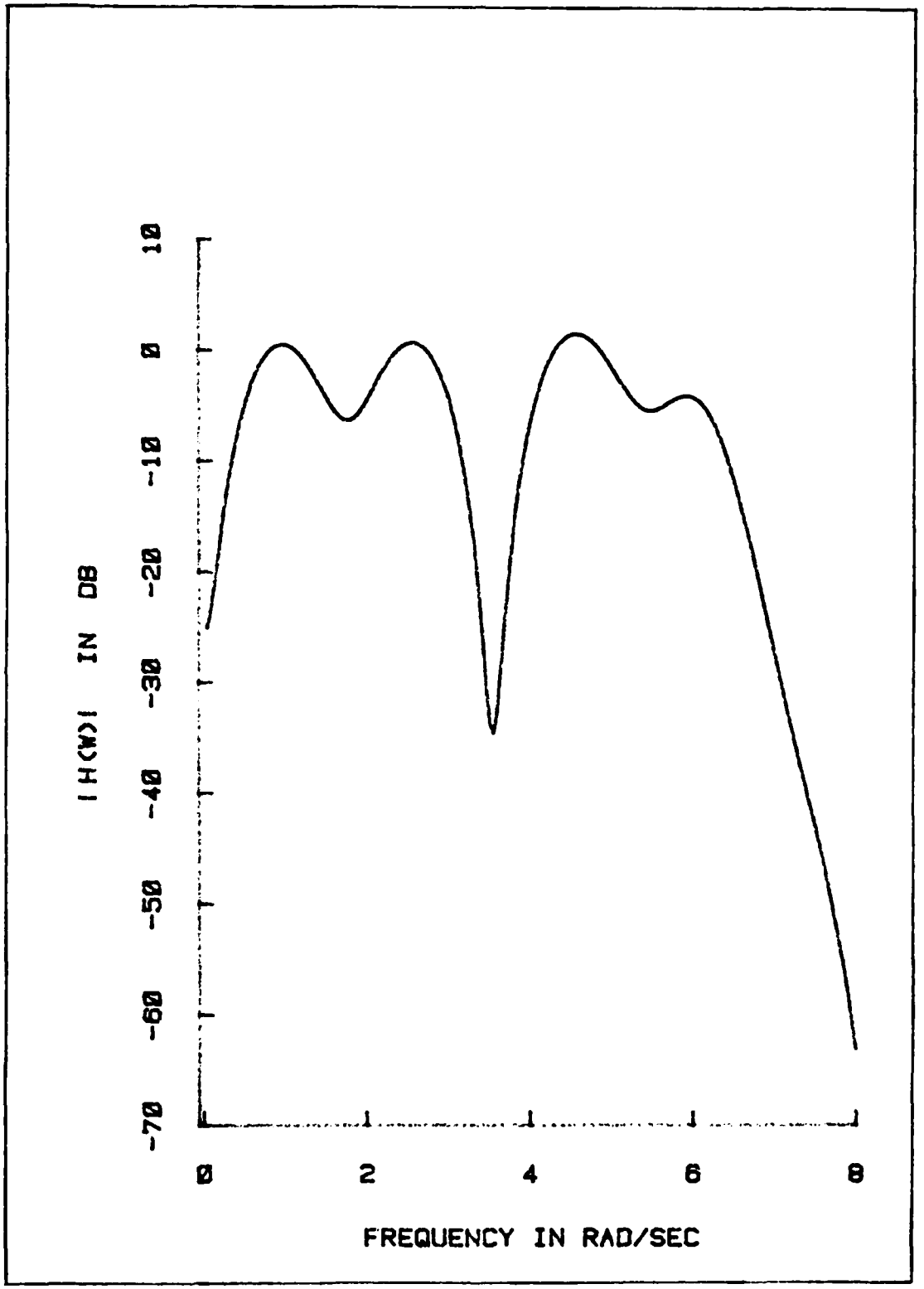


Figure V-12 200 Iterations

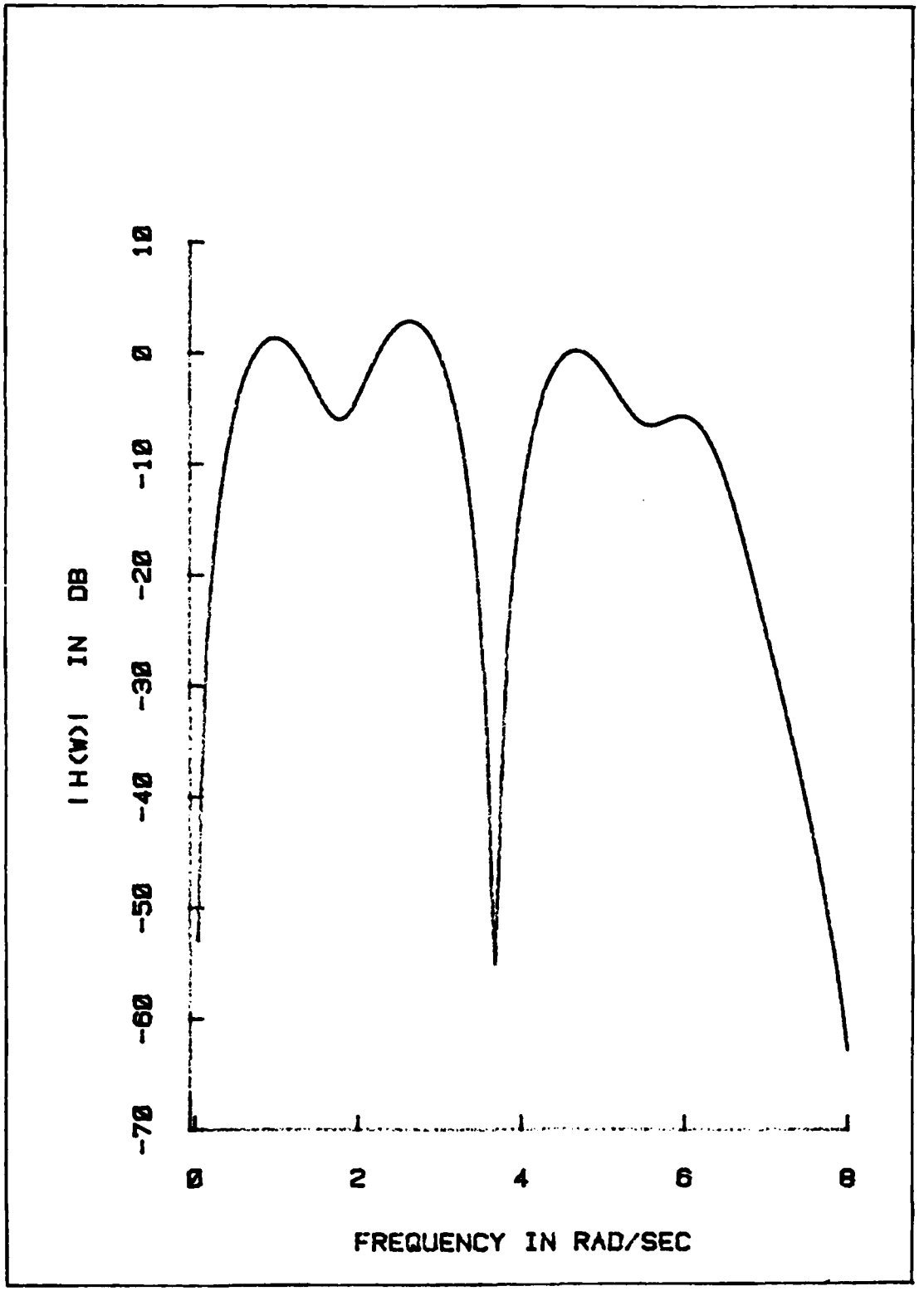


Figure V-13 500 Iterations

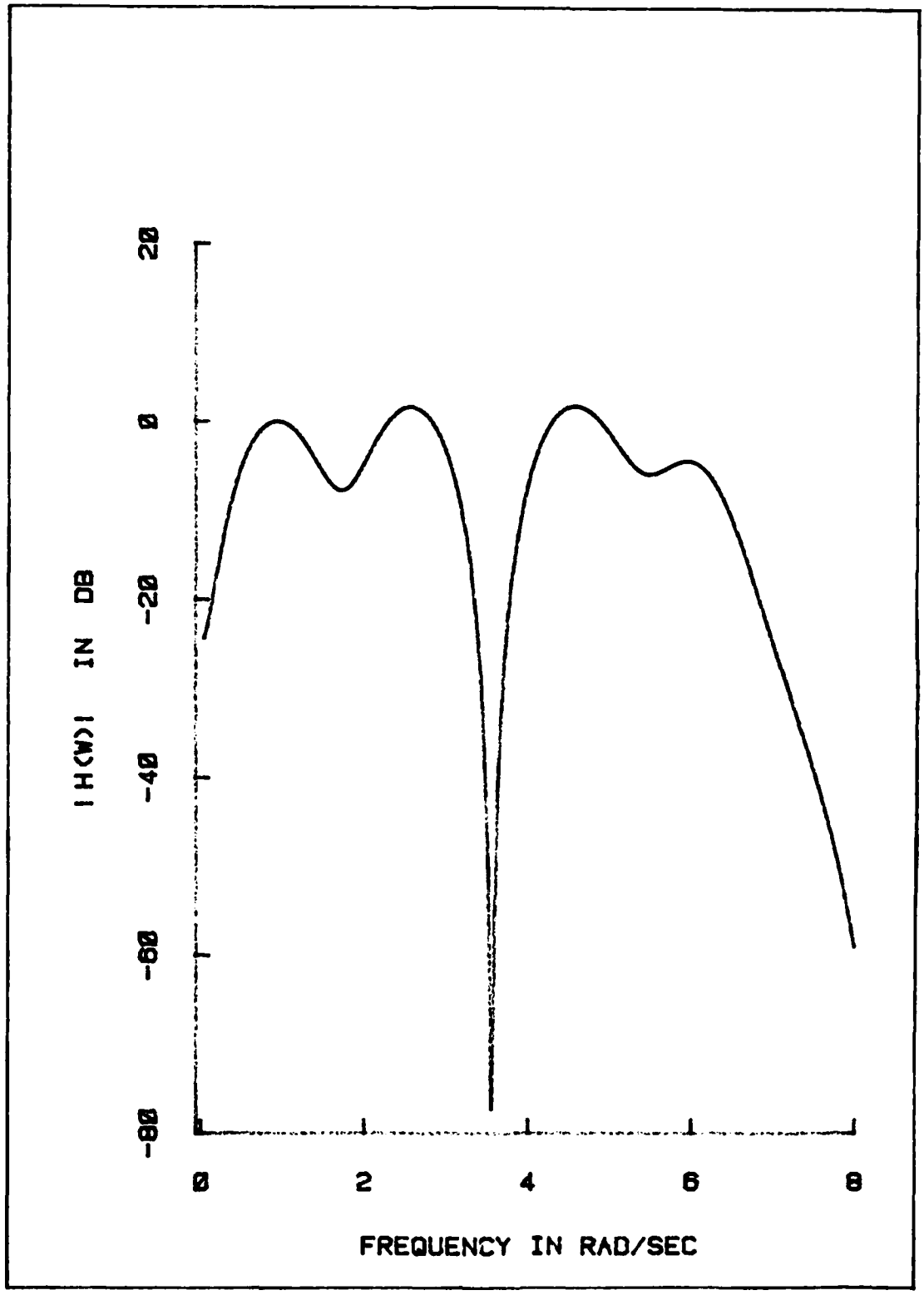


Figure V-14 1000 Iterations

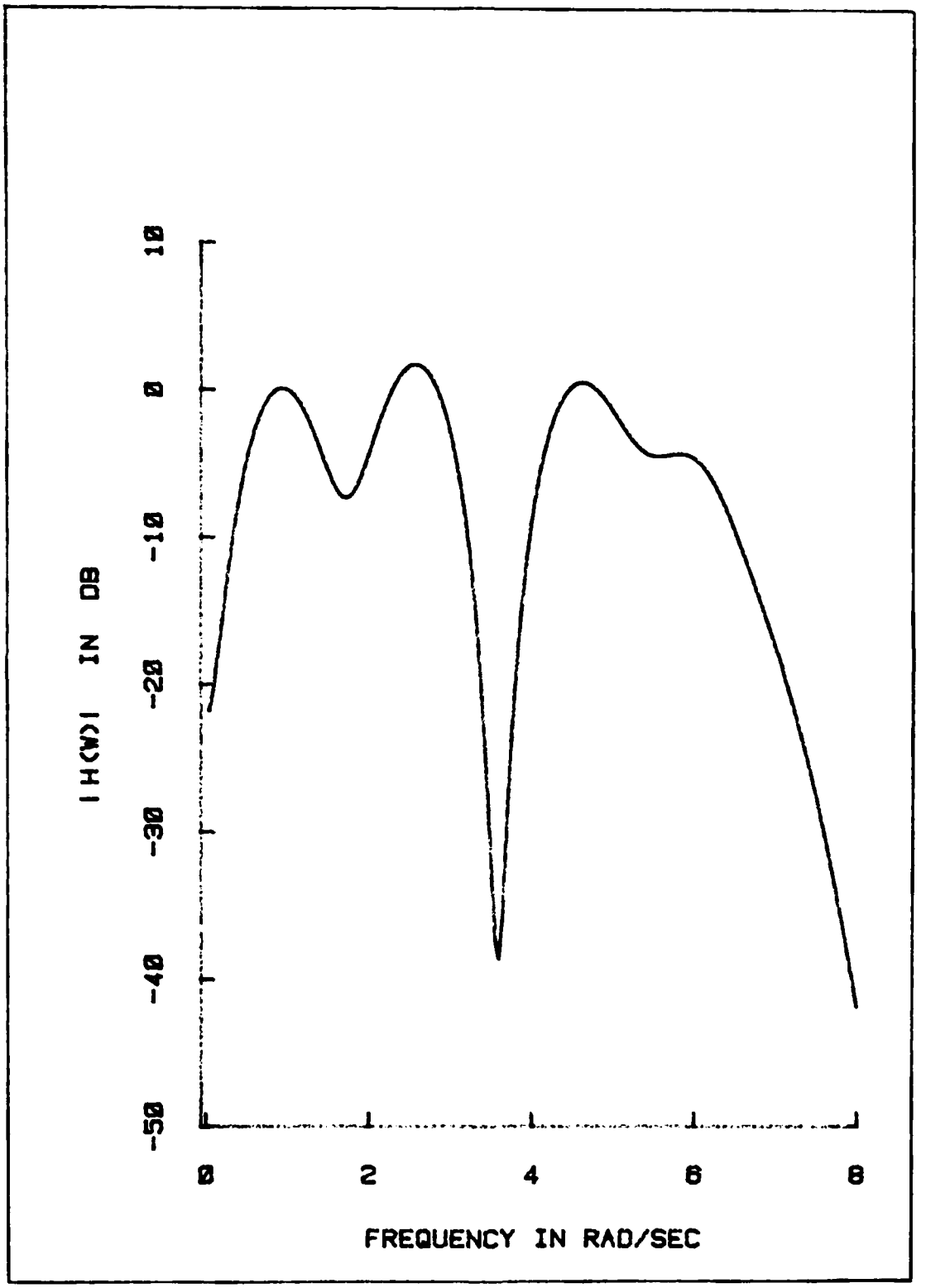


Figure V-15 2000 Iterations

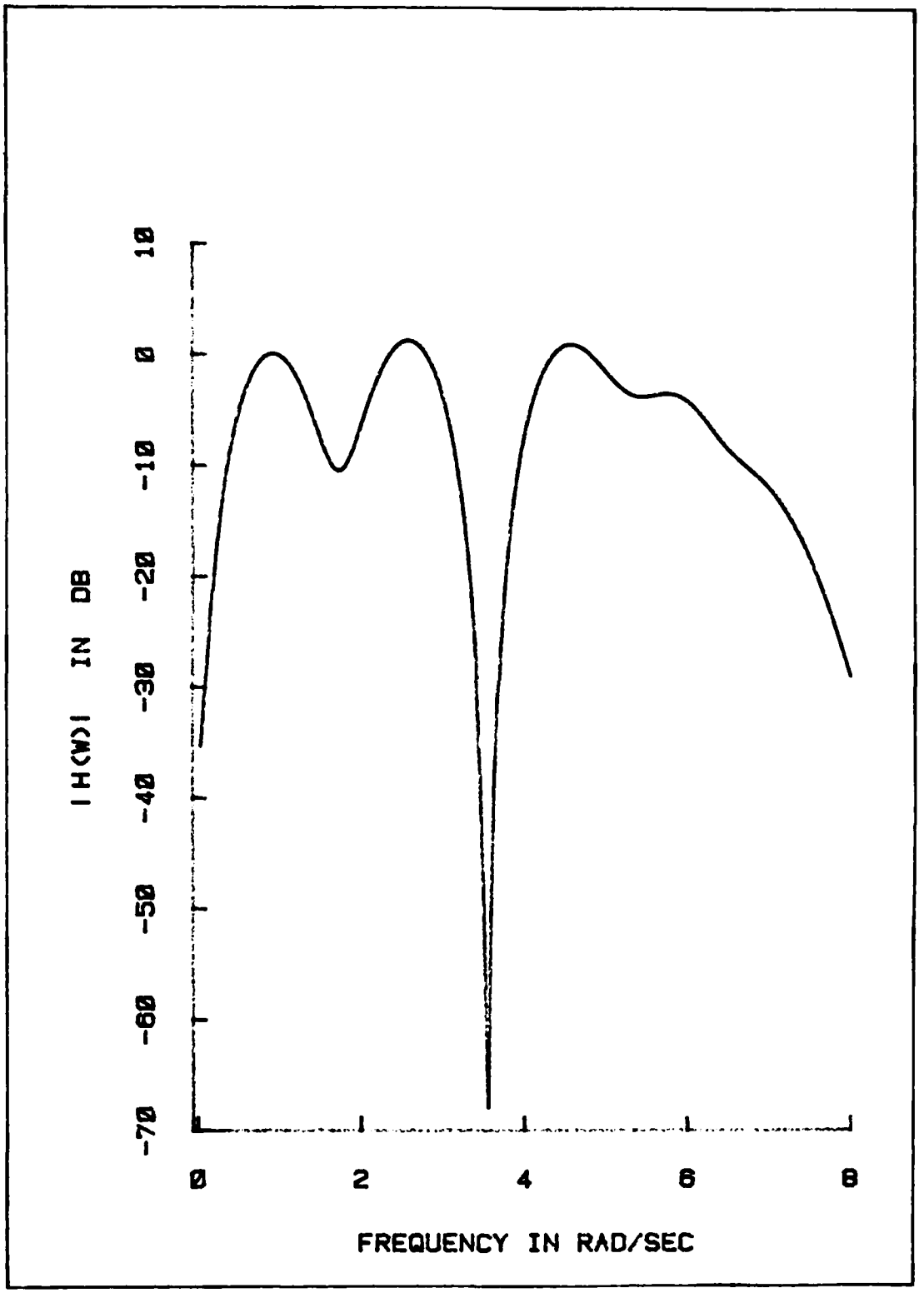


Figure V-16 5000 Iterations

least -34dB which would substantially reduce the effect of a 20dB jammer.

The above results, (Figures V-4 to V-16) were obtained with the B matrix weighting values for all the constraint frequencies selected as 10.0. The ability to easily change the weights associated with each constraint frequency makes this adaptive filter particularly useful.

For example, Figure V-17 shows the transfer function after 500 iterations with a jammer frequency of 4.7 rad/sec and all the frequency constraint weights set to 10.0. However, suppose that by prior knowledge (or through external circuitry within a few data samples) it is known that the jammer's frequency will be (or is) 4.7 rad/sec. By changing the B matrix weights associated with 4.2 and 5.2 rad/sec to 1.0 and keeping the other weights equal to 10.0 the improvement shown by Figure V-18 is obtained.

Frequency Hopping Jammer

This section illustrates how well this adaptive filter adapts when the jammer hops to a new frequency. If the jammer determines or senses that his jamming at one frequency is not effective he may "hop" his jamming signal to another frequency. It is important that the adaptive filter quickly recognize the new jamming frequency and adjust its notch accordingly.

Start with a well established (500 data samples) jammer at 5.7 rad/sec illustrated by Figure V-19. The jammer then hops to 4.7 rad/sec. After just 20 more data samples or

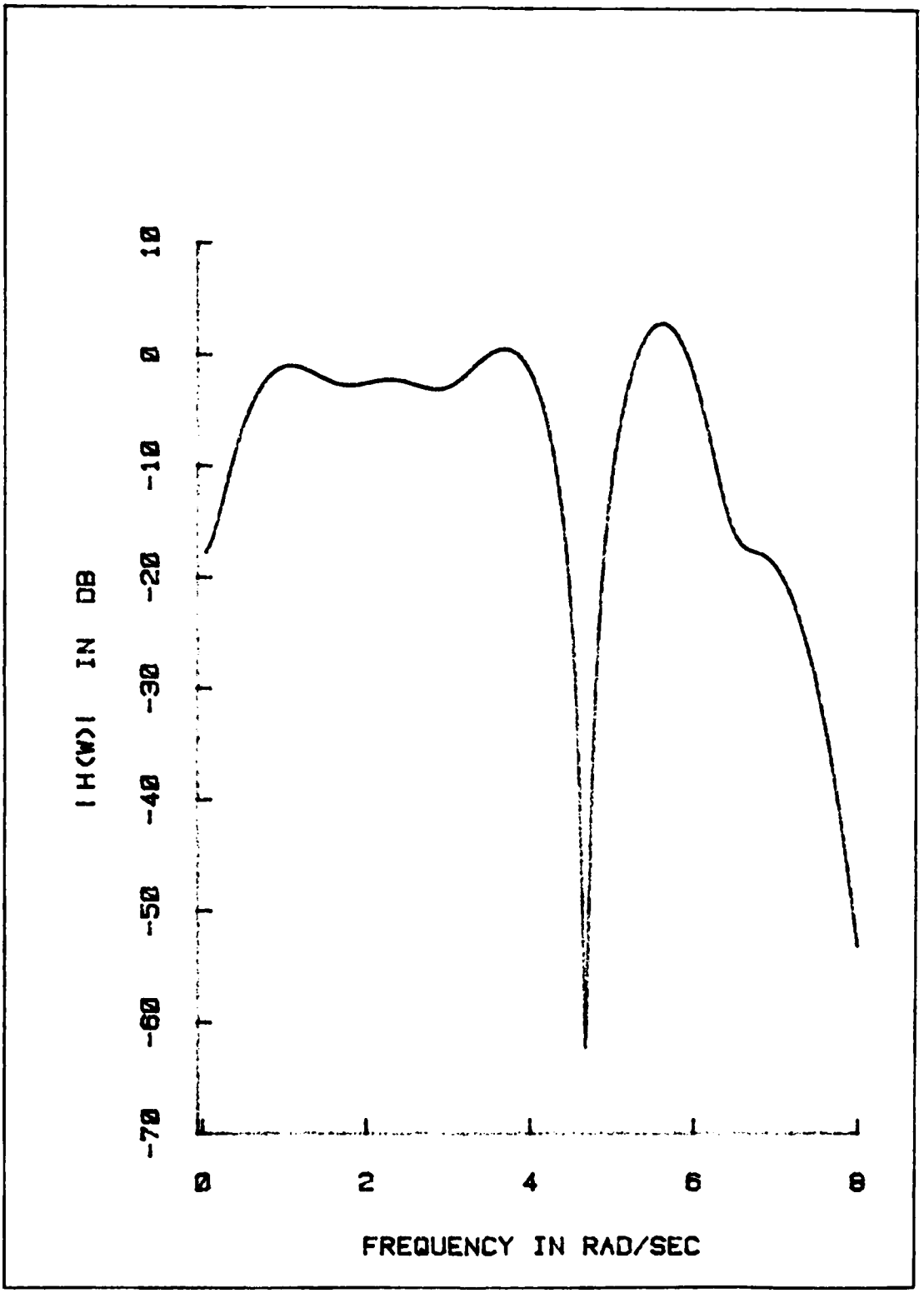


Figure V-17 500 Iterations

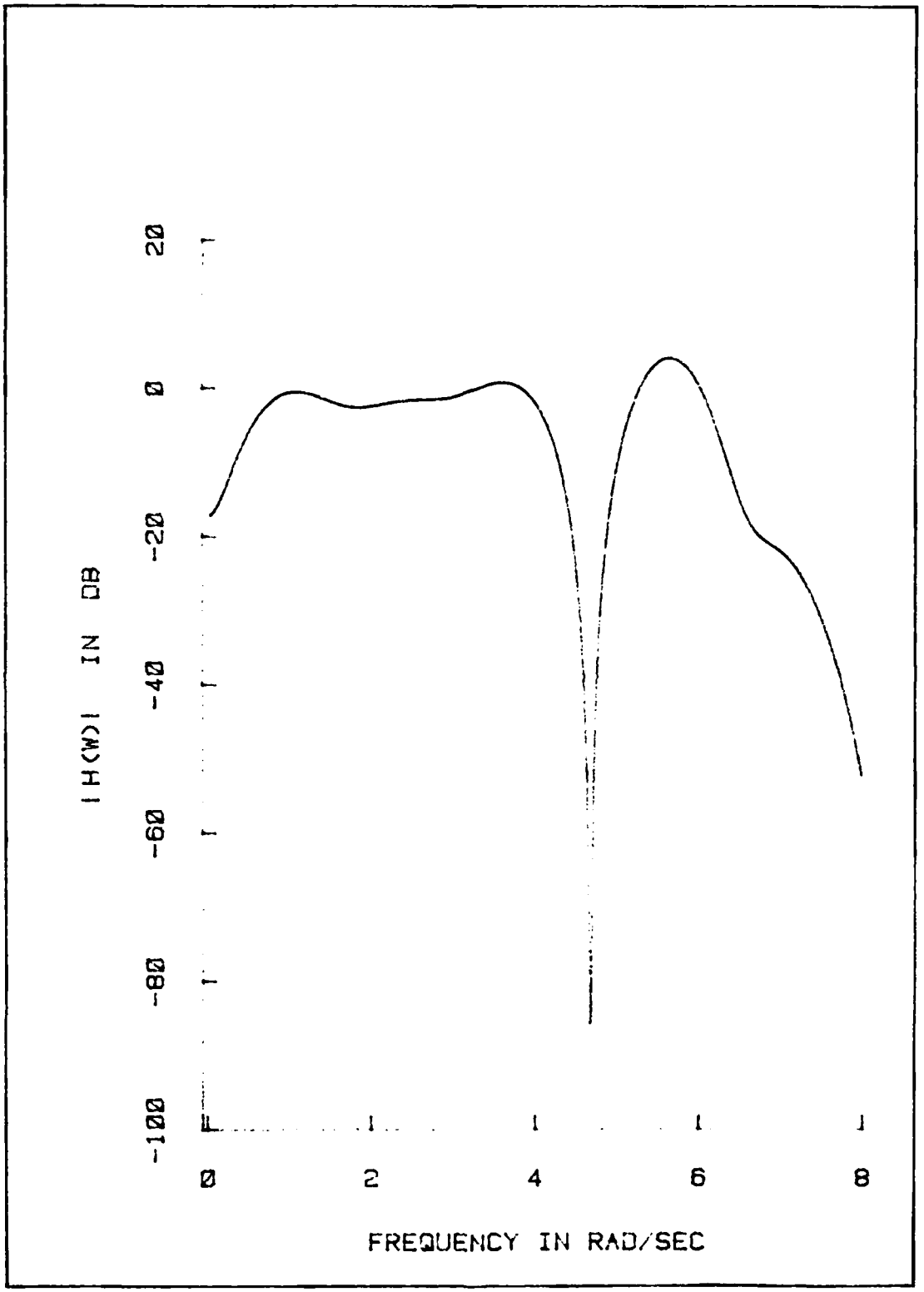


Figure V-18 500 Iterations

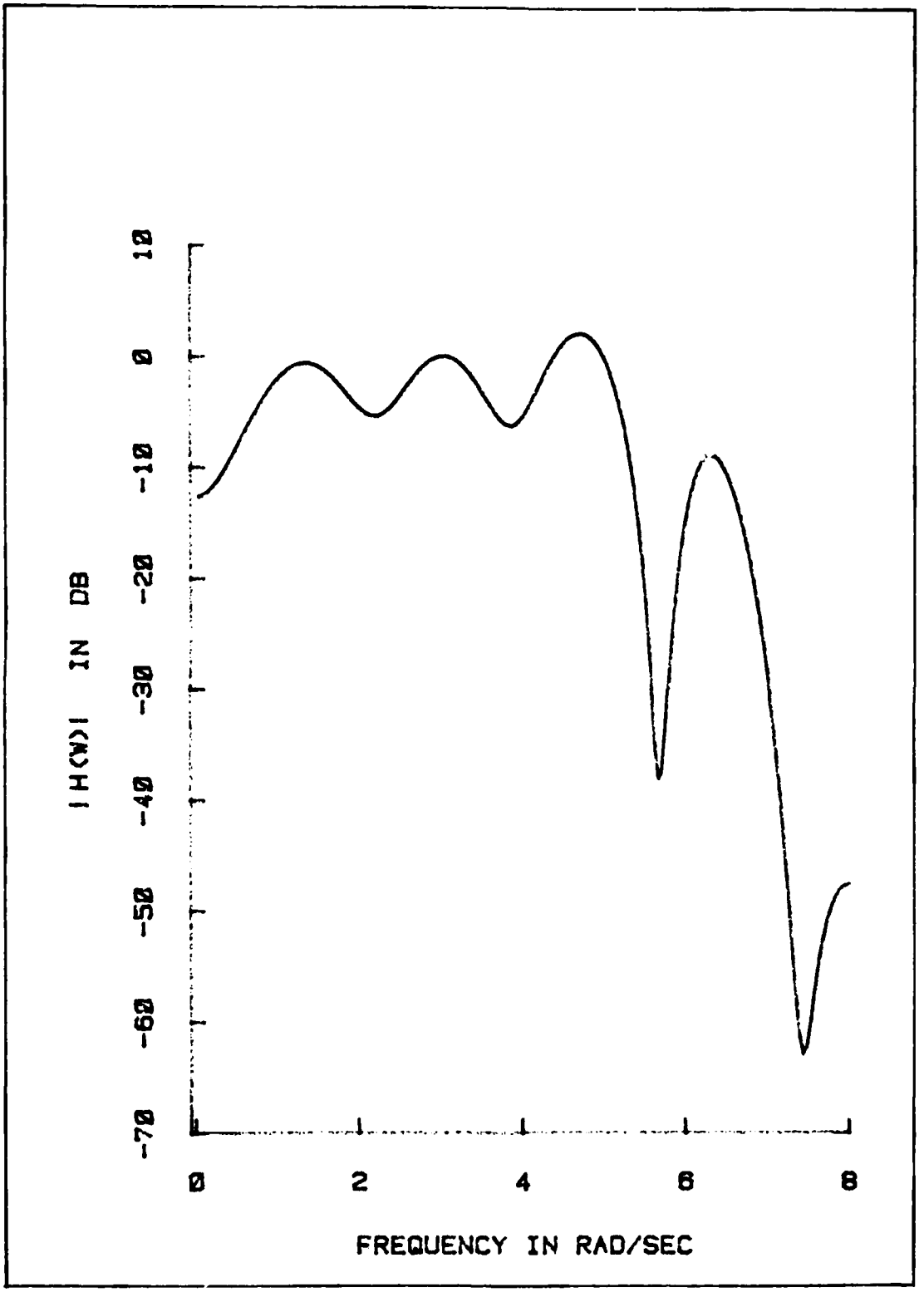


Figure V-19 500 Iterations

iterations the adaptive filter has produced an effective notch at 4.7 rad/sec as shown by Figure V-20.

As a second example, start again with the jammer at 5.7 rad/sec (Figure V-19). This time the jammer hops to 1.6 rad/sec. As shown in Figure V-21, after only 10 data samples, the notch is beginning to form and after 40 data samples Figure V-22 shows an extremely effective notch at 1.6 rad/sec.

Two Jammers

This section shows how well the adaptive filter responds to two jammers at different frequencies. Suppose, that the jammer determines that hopping his jamming signal is not effective. The jammer may decide to divide his available power between two jamming signals. Thus, it is important that the adaptive filter be effective for more than one jammer. All the results of this section involve two jamming signals with powers of 50 watts or 17 dB each.

In the first example, Figure V-23 shows the transfer function of the adaptive filter after 200 iterations with jammers located at 3.7 and 4.7 rad/sec. In the second example, the jammers are located at 1.6 and 5.7 rad/sec, and after 100 iterations, the transfer function appears as shown in Figure V-24 with all the frequency constraint weights set to 10.0. To improve this transfer function, suppose, as was done previously, that by prior knowledge or through additional circuitry within a few data samples it is known that the jammers' frequencies are 1.6 and 5.7 rad/sec. By

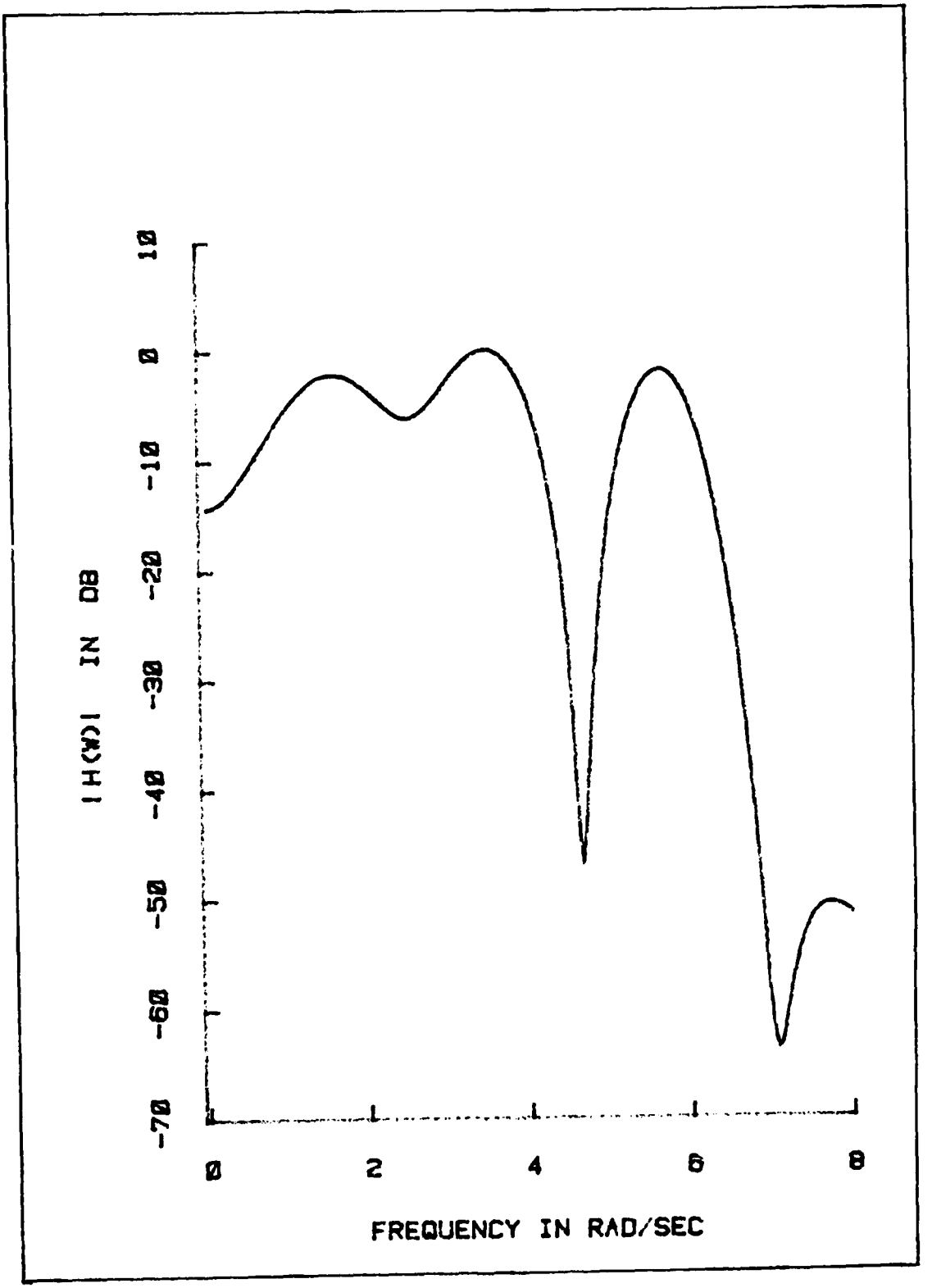


Figure V-20 20 More Iterations

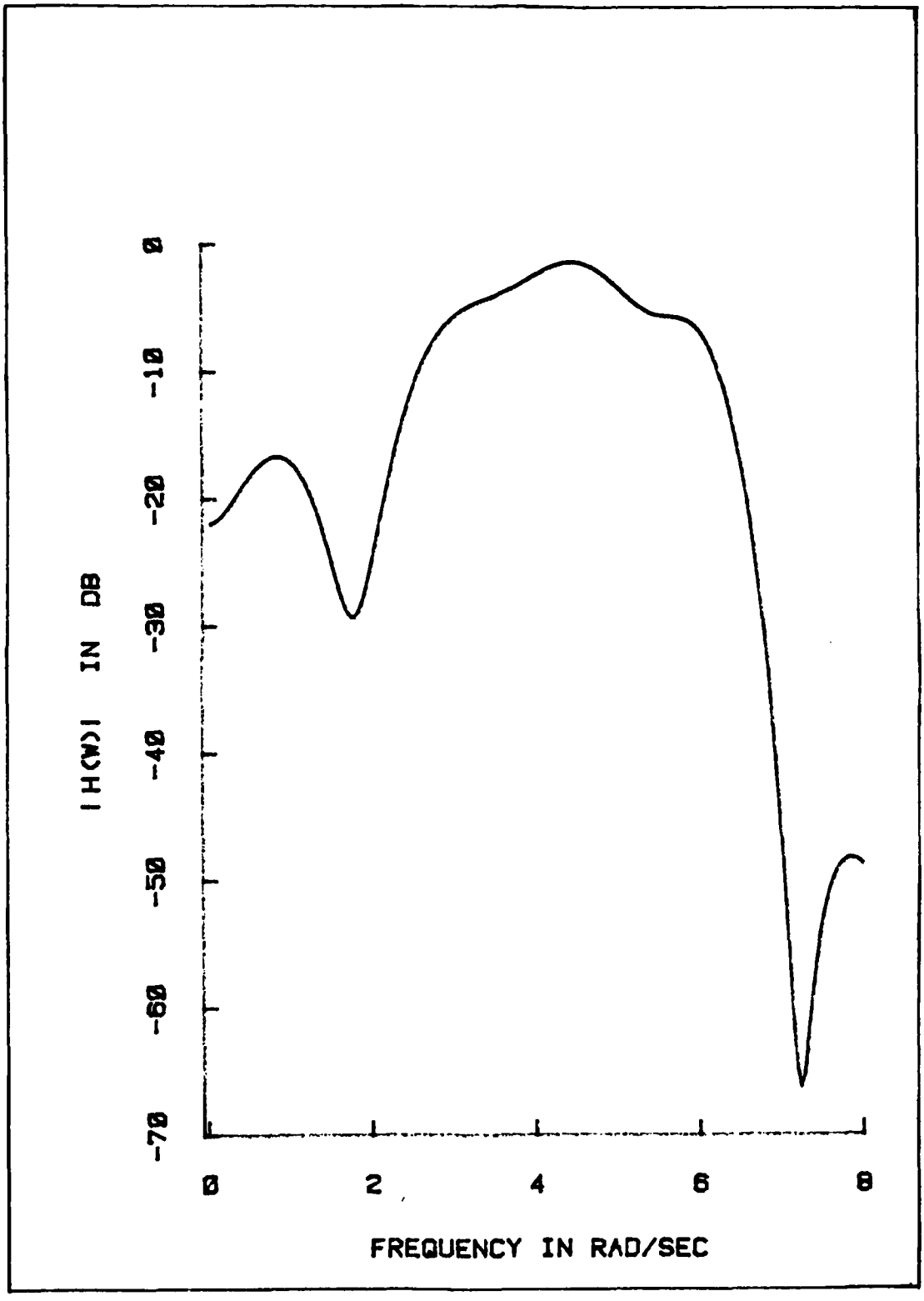


Figure V-21 10 More Iterations

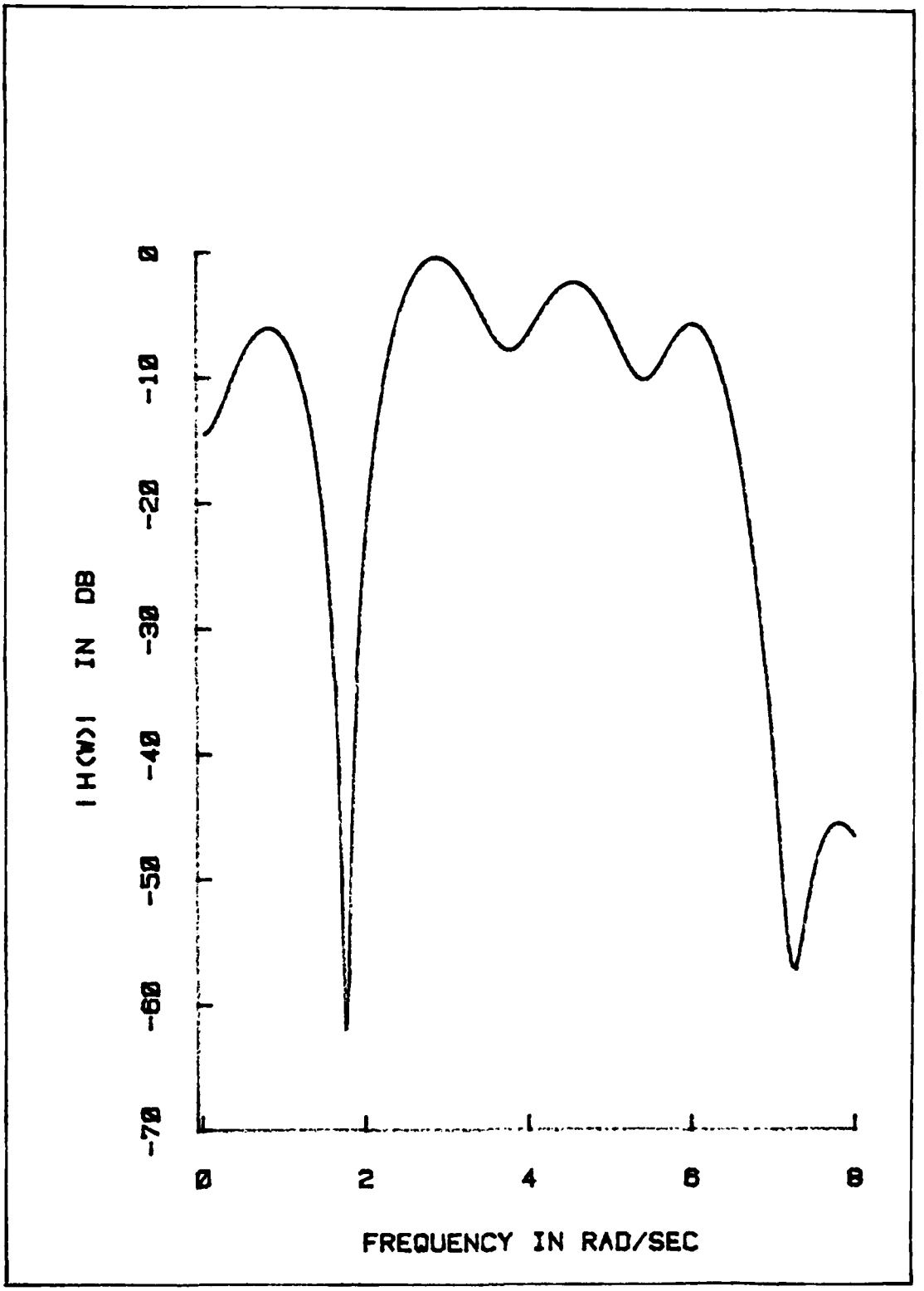


Figure V-22 40 Iterations

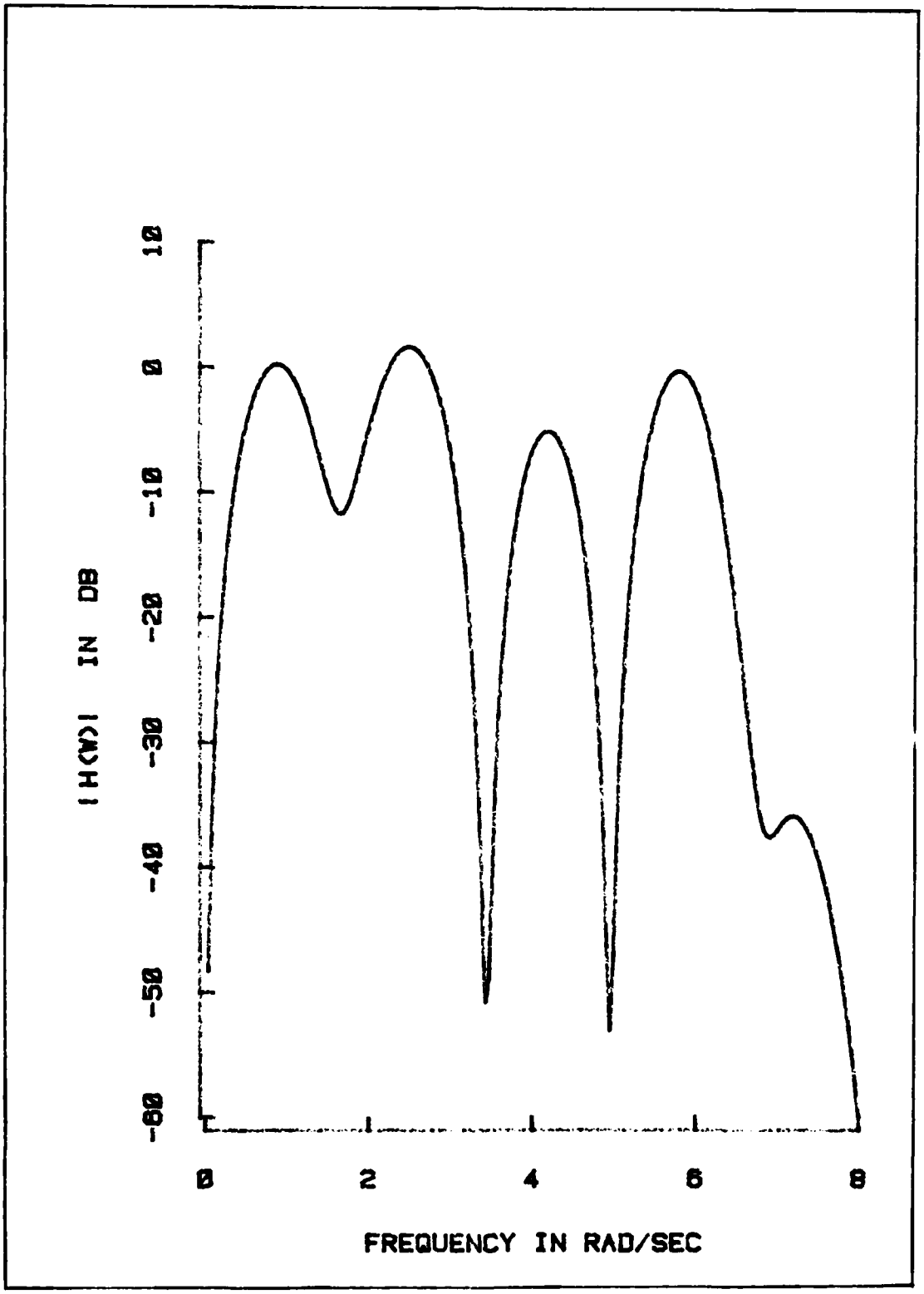


Figure V-23 2 Jammers

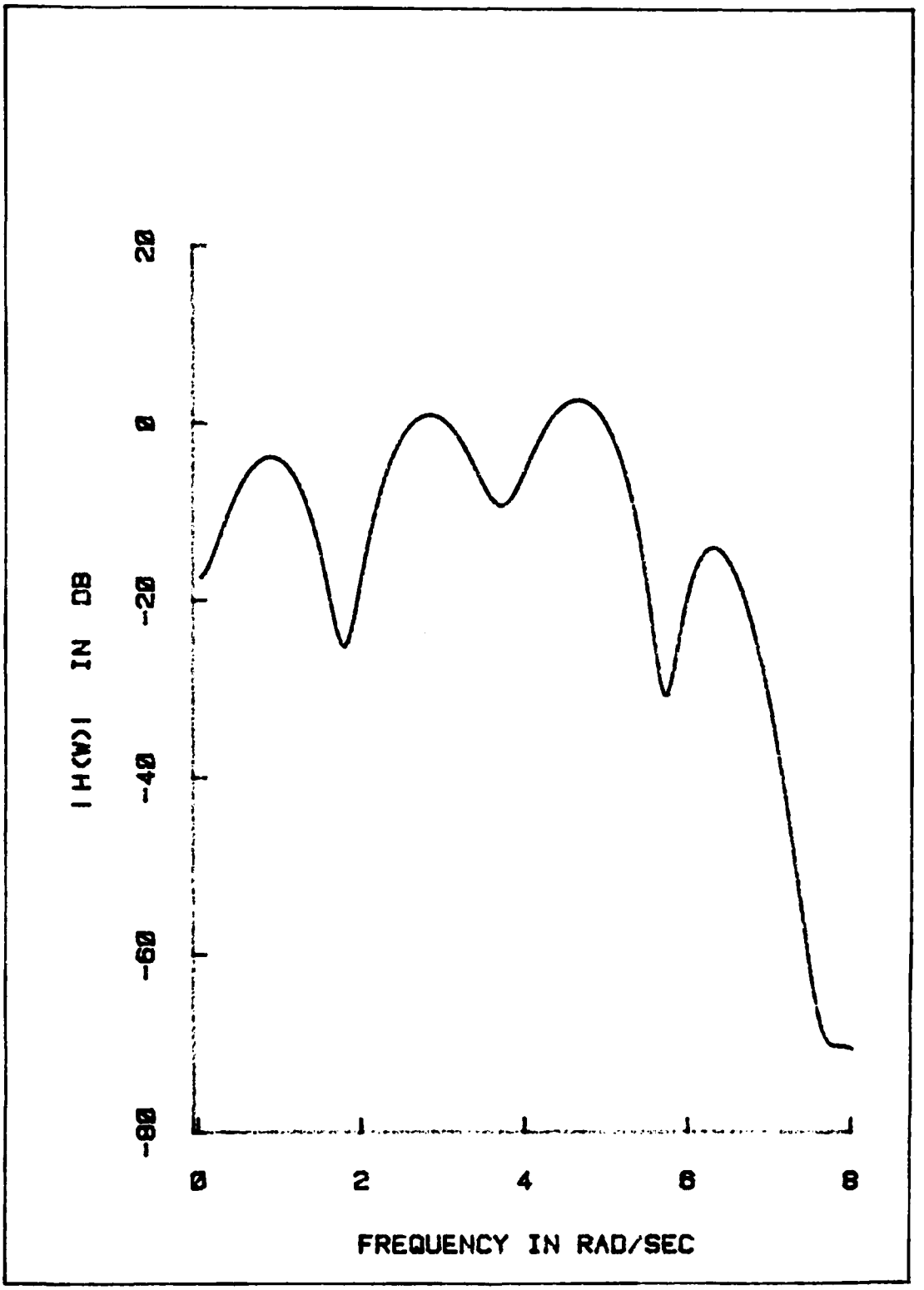


Figure V-24 2 Jammers

keeping the B matrix weights associated with 3.1 and 4.2 rad/sec set to 10.0 and changing all the other weights to 1.0, the improvement shown by Figure V-25 is obtained.

Performance Improvement

This section presents the improvement in signal to noise type ratios measured at the output of a spread spectrum receiver preceded by the new adaptive filter. The powers of the various signals were calculated as explained in Chapter IV under the "Subroutines" section. Several examples are given.

When the adaptive filter algorithm runs for 49 iterations as shown in Figure V-9 then a notch depth of -44dB is obtained which appears to be an average depth. When using an SS receiver with a processing gain of 100, the following results are obtained:

S/N = 43.2

S/J = 4.27

S/(N+J) = 3.88

When the adaptive filter is not used with the above receiver these results are obtained:

S/N = 57.5

S/J = 1.0

S/(N+J) = 0.983

As a second example, a greater improvement in S/(N+J) can be found by running the algorithm for only 35 iterations to obtain the deeper notch shown in Figure V-7. Using this result and the same SS receiver gives:

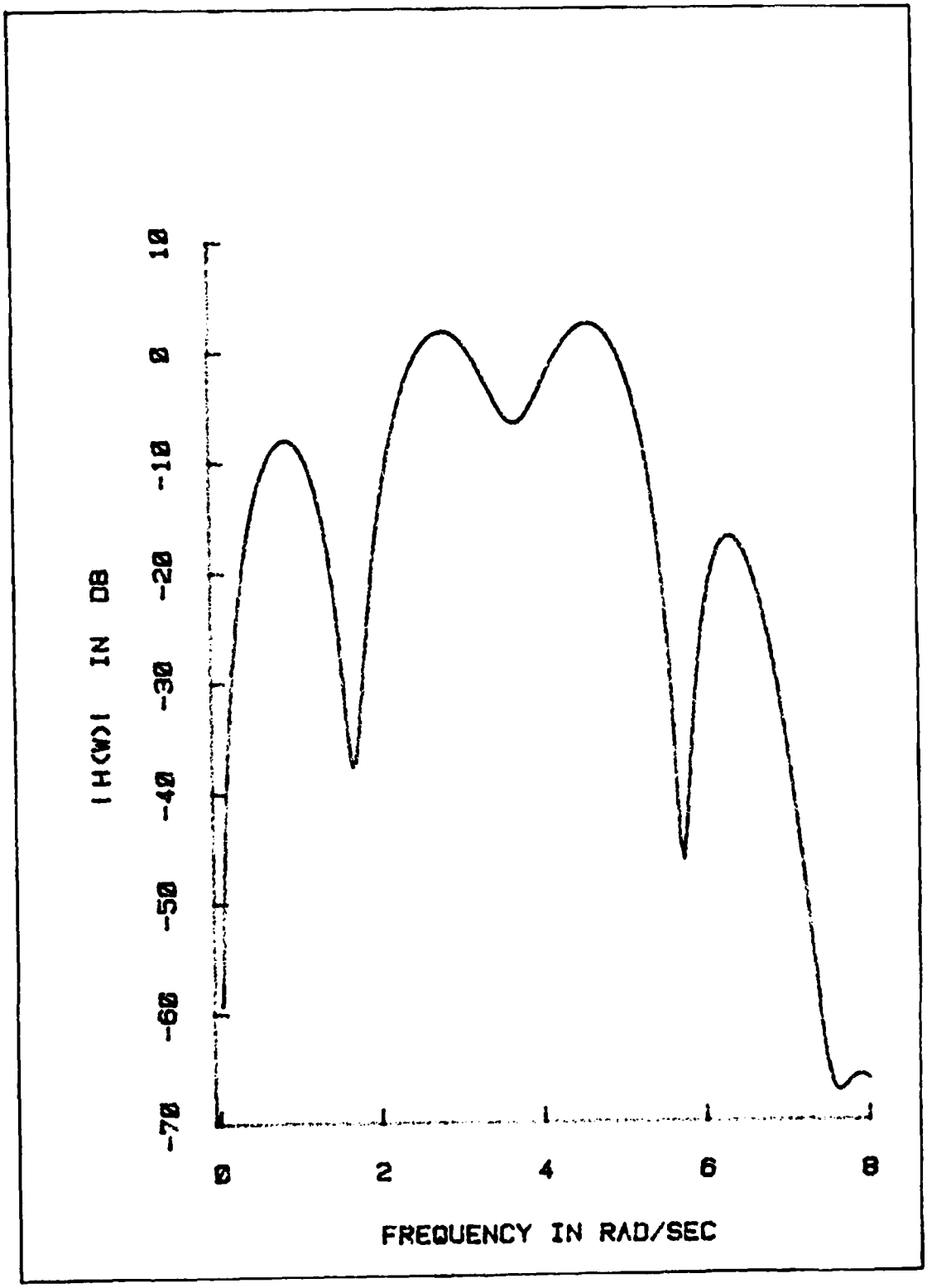


Figure V-25 2 Jammers

S/N = 57.1

S/J = 6.13

S/(N+J) = 5.53

Thirdly, when using an SS receiver with a PG of 1000 and running the algorithm for 500 iterations (Figure V-17) the results are:

S/N = 290.9

S/J = 57.6

S/(N+J) = 48.1

Finally, recall that the notch in Figure V-17 was improved (Figure V-18) by changing the appropriate frequency constraint weights. This change also improves the receiver performance results as shown by:

S/N = 294.8

S/J = 69.4

S/(N+J) = 56.2

These results are summarized in the final chapter, Conclusions and Recomendations.

VI. CONCLUSIONS AND RECOMMENDATIONS

Conclusions

This thesis has determined the improvement in perforance gained by using the soft-constraint LMS algorithm with an adaptive filter in conjunction with a spread spectrum receiver. For an average size notch depth in the adaptive filter it was found that the signal to jammer ratio improved by a factor of 4.27 over an SS system without an adaptive filter. For an extremely deep notch the signal to jammer ratio improved by a factor of 6.13.

It was also found that by appropriate choice of the frequency constraint weights a slightly greater signal to jammer ratio can be obtained.

More importantly, it was found that the soft-constraint LMS algorithm is well suited for use in adaptive notch filters. In the presence of noise and SS signals, a 20 db jamming signal causes notch depths ranging from -34 dB to -77 dB.

Recommendations

Much more information about the soft-constraint LMS algorithm/adaptive filter/SS receiver combination can be obtained with no changes made to the ADAPSS program. It is therefore recommended that

(1) the effect of higher and lower power jammers be studied,

(2) the effect of higher and lower noise powers be studied,

(3) the effect of larger frequency constraint weights be studied,

(4) the optimum convergent factor be found. (It is suggested that finding the maximum eigenvalue of the data may be helpful here.)

It is further recommended that by making minor changes to the program, studies may be made to find

(5) the effect of changes in the sampling time,

(6) the effect of different constraint frequencies,

(7) the effect of a slowly moving or wideband jammer.

Of course, with extensive program changes the effect of more or less taps in the adaptive filter can be studied and even different adaptive filters with the same SS receiver could be compared.

Bibliography

1. Larsen, MGen Doyle E. "C^3CM: Lessons Learned; Where Do We Go From Here?" Signal, 37 (8): (April 1983).

2. "Frequency Agile HAVE QUICK System Nears Completion," Defense Electronics, 14 (9): 51-52 (September 1982).

3. Robinson, Clarence A.,Jr. "Common Armed Forces Link Sought," Aviation Week and Space Technology, 117 (20): 51-58 (November 15, 1982).

4. Bolen, Nelson E. "JTIDS Coordinates Tactical Elements," Defense Electronics, 14 (1): 104-105 (January 1982).

5. Cuccia, C. Louis. "Spread Spectrum Systems Serve Nearly All C^3 Aspects," Microwave System News, 78-91 (April 1982).

6. Klass, Philip J. "Advanced Satcom Designed for Military," Aviation Week and Space Technology, 115 (13): 72-74 (September 28, 1981).

7. Batson, B.H. and G.K. Huth. "Spread Spectrum Techniques for the Space Shuttle," Proceedings of the 1979 IEEE National Telecommunications Conference, 54.41-54.42 (1979).

8. Elnoubi, Said and S.C. Gupta. "Performance of Frequency Hopped MSK Spread Spectrum Mobile Radio System with Discriminator Detection," Proceedings of the 1981 IEEE National Telecommunications Conference. G3.3.1-G3.3.2. (1981).

9. Pickholtz, Raymond L., et al. "Theory of Spread Spectrum Communications - A Tutorial," IEEE Transactions on Communications, COM-30 (5): 855-884 (May 1982).

10. McCool, John M. and Bernard Widow. "Principles and Applications of Adaptive Filters: A Tutorial Review," Proceedings of the 1980 IEEE International Symposium on Circuits and Systems, 1143-1157.

11. Ketchum, John W. and John G. Proakis. "Adaptive Algorithms for Estimating and Suppressing Narrow-Band Interference in PN Spread-Spectrum Systems," IEEE Transactions on Communications, COM-20 (5): 913-923 (May 1982).

12. Li, Loh-Ming and Laurence B. Milstein. "Rejection of Narrow-Band Interference in PN Spread-Spectrum Systems Using Transversal Filters," IEEE Transactions on Communications, COM-30 (5): 925-928 (May 1982).

13. Hsu, Frank M. and Arthur A. Giordano. "Digital Whitening Techniques for Improving Spread Spectrum Commuications Performance in the Presence of Narrowband Jamming and Interference," IEEE Transactions on Communications, COM-26 (2): 209-216 (February 1978).

14. Mostafa, Abd El-Samie, et al. "Improvements of Antijam Performance of Spread-Spectrum Systems," IEEE Transactions on Communications, COM-31 (6): 803-808 (June 1983).

15. Proakis, John G. Digital Communications. New York: McGraw-Hill Book Company, 1983.

16. Ayala, I.L. "On a New Adaptive Lattice Algorithm for Recursive Filters," IEEE Transactions on Acoustics, Speech,and Signal Processing, ASSP-30 (2): 316-319 (April 1982).

17. Friedlander, Benjamin. "Lattice Filters for Adaptive Processing," Proceedings of the IEEE, 70 (8): 829-866 (August 1982).

18. ----- "A Recursive Maximum Likelihood Aogorithm for ARMA Line Enhancement," IEEE Transactions on Acoustics, Speech, and Signal Processing, ASSP-30 (4): 651-67 (August 1982).

19. Friedlander,Benjamin and Martin Morf. "Least Squares Algorithm for Adaptive Linear-Phase Filtering," IEEE Transactions on Acoustics, Speech, and Signal Processing, ASSP-30 (3): 381-390 (June 1982).

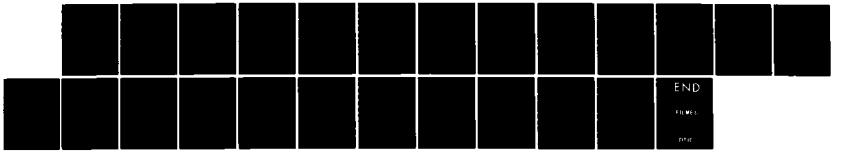
20. Widrow, B., R. Chestek, H. Mesiwab, and K. Duvall. Research on Large Adaptive Arrays. RADC-TR-79-308. Griffiss Air Force Base, NY: Rome Air Development Center, December 1979. (AD-A 081 388).

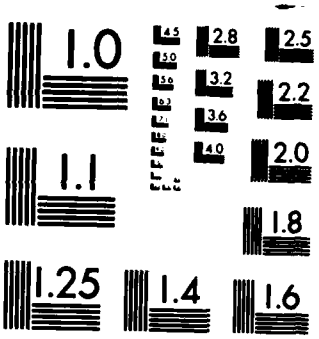
21. Torrieri, Don J. Principles of Military Communication Systems. Dedham, Mass: Artech House, Inc., 1981.

22. Liu, Gerald C. and John F. Rowe, Jr. "A Computer Simulation of a Jammed Direct Sequence Spread Spectrum Communications Link," Proceedings of the 1979 National Telecommunications Conference. 15.6.1-15.6.5 (1979).

AD-A151 890 THE PERFORMANCE OF A NEW ADAPTIVE FILTER FOR DIGITAL COMMUNICATIONS(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERING J R WAY 2/2

UNCLASSIFIED JUN 84 AFIT/GE/EE/84S-12 F/G 17/2 NL





23. Spellman, Marc. "Spread Spectrum Radios Thwart Hostile Jammers," Microwaves, 20 (9): 85-90 (September 1981).

24. Viterbi, Andrew J. "Spread Spectrum Communications - Myths and Realities," IEEE Communications Magazine, 17 (3): 11-18 (May 1979).

25. Perry, Bernard D. "A New Wideband HF Technique for MHz-Bandwidth Spread-Spectrum Radio Communications," IEEE Communications Magazine, 21 (6): 28-36 (September 1983).

26. Cook, Charles E. and Howard S. Marsh. "An Introduction to Spread Spectrum," IEEE Communications Magazine, 21 (2): 8-16 (March 1983).

27. Spellman, Marc. "A Comparison Between Frequency Hopping and Direct Spread PN as Antijam Techniques," IEEE Communications Magazine, 21 (2): 37-51 (March 1983).

28. Sass, Paul F. "Why is the Army Interested in Spread Spectrum?" IEEE Communications Magazine, 21 (4): 23-25 (July 1983).

29. Utlaut, William F. "Spread Spectrum-Principles and Possible Application to Spectrum Utilization and Allocation," IEEE Communications Magazine, 16 (5): 21-30 (September 1978).

30. Ristenbatt, Marlin P. "Alternative in Digital Communications," Proceedings of the IEEE, 61 (6): 703-721 (June 1973).

31. Dixon, R.C. Spread Spectrum Systems. New York: John Wiley & Sons, Inc., 1976.

32. Dixon, Robert C. "Why Spread Spectrum?" IEEE Communications Society Digest, 13 (4): 21-25 (July 1979).

33. Dixon, Robert C. Spread Spectrum Technique. New York: IEEE Press, 1976.

34. Shepard, Michael M. "The Performance of a PN Spread Spectrum Receiver Preceded by an Adaptive Interference Suppression Filter." Unpublished MS thesis. School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, December 1982.

35. Papoulis, Athanasios. Probability, Random Variables, and Stochastic Processes. New York: McGraw-Hill Book Company, 1965.

36. Peebles, Peyton Z., Jr. Communication System Principles. Reading,Mass: Addison-Wesley Publishing Company,1976.

37. Hall, Thomas J. "Intercept Vulnerability of Direct Sequence Psuedo-Noise Encoded Spread Spectrum Waveforms," Unpublished MS thesis. School of Engineering, Air Force Insitute of Technology, Wright-Patterson AFB, Ohio, December 1979.

38. Melsa, James L. and David L.Cohn. Decision and Estimation Theory. New York: McGraw-Hill Book Company, 1978.

39. Widrow, Bernard, John R. Glover, Jr., John M.McCool, et al., "Adaptive Noise Cancelling: Principles and Applications," Proceedings of the IEEE, 63 (12): 1692-1716 (December 1975).

40. Widrow, Bernard and John M. McCool. "A Comparison of Adaptive Algorithms Based on the Methods of Steepest Descent and Random Search," IEEE Transactions on Antennas and Propagation, 24 (5): 615-637 (September 1976).

41. Widrow, Bernard. "Adaptive Filters," Aspects of Network and System Theory, edited by R.E. Kalman and N. DeClaris. New York: Holt, Rinehart and Winston, 1971.

42. Syed, Vaqar H. Lecture materials entitled, "Adaptive Antenna Arrays," distributed in EE 6.73, Applications of Communications Technology. School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, Fall 1983.

43. Antoniou, Andreas. Digital Filters: Analysis and Design. New York: McGraw-Hill Book Company, 1979.

44. Oppenheim, Alan V. and Ronald W. Schafer. Digital Signal Processing. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1975.

Appendix A

ADAPSS Program

The ADAPSS (Adaptive/Spread Spectrum) program that follows was written in FORTRAN 77 using the UNIX "vi" editor and executed on a VAX 11/780. The program results are stored in various files for subsequent plotting. The results presented in Chapter V were obtained by using the "S" language and an HP plotter. Details on the execution of the ADAPSS program can be found in Chapter IV.

Input File

RNFILE - This file of random numbers with a normal (0,1) distribution must reside in memory before execution of the ADAPSS program begins. The generation of this file is covered in the "Preliminary Steps" section of capter IV.

Output Files

wvalues - frequency values in radians used to plot the adaptive filter's frequency response after completion of program.

h1values - magnitude in dB corresponding to the frequencies in the wvalues file. The 1 indicates these results are for the first position of a hopping jammer.

h2values - magnitude in dB for the second position of a hopping jammer.

h3values - magnitude in dB for the third position of a hopping jammer.

Double Precision Variables

X - data value which enters the adaptive filter.

Y - data value which exits the adaptive filter.

S - spread spectrum signal value.

J - jammer value.

N - noise value.

RN - random number read from RNFILE.

T - sampling time.

AMP - amplitude of noise spectral shaping filter.

NPWR - noise power.

JPWR - jammer power.

FJ - frequency of jammer.

PG - processing gain.

TAU - this scaling factor needs to be set to 0.05 in the initial conditions.

WT - adaptive filter weight vector.

IDENT - identity matrix.

TERM1, TERM2, TERM3, TERM4 - intermediate results.

A, B, H - "A", "B", and "H" matrices of the soft-constraint algorithm.

AT - "A" matrix transposed.

ATB, ATBA, ATBH - intermidiate matrix results.

WC - array of constraint frequencies.

KS - convergent factor.

PI - 3.1415926536.

Integer Variables

K - main do loop counter.

KMAX - user selected value of final do loop iteration.

CASE - user selected value outputted in results file .

Q, L, M1 - do loop counters.

FREQ - do loop counter used to keep track of hops made by jammer.

M1 - user selected number of frequencies at which jammer will appear.

Real Variable

XK - the integer K converted to a real number.

Subroutine XFER

Complex Variables

HXFER - complex magnitude of transfer function.

HXFER1, HXFER2 - intermediate results.

I - imaginary number (0+j1).

LPD - denominator of low pass filter.

Double Precision Variables

WT - adaptive filter weight vector.

W - frequency in radians.

T - sampling time.

ABSH - absolute magnitude of transfer function.

DELW - an increment of frequency.

DB - magnitude of transfer function in dB.

Real Variables

PI - 3.14159265

PI2 - 2*PI

A1 thru A9 - constants used in calculating the 10th order low pass filter.

LPH - magnitude of low pass filter.

LPNR - numerator of low pass filter.

LPDR - magnitude of low pass filter denominator.

Integer Variables

L - do loop counter.

FREQ - do loop counter used to keep track of number of hops made by jammer.

Subroutine PWROUT

Real Variables

YHAT - data value at output of adaptive filter.

NOPOUT - noise power at filter output.

SIPOUT - signal power at filter output.

JAPOUT - jammer power at filter output.

NOISIN - calculated input noise power.

PWR, NCALC - intermediate results.

```
      PROGRAM ADAPSS
*************************************************************************
*       Program:    ADAPSS                                              *
*       Author :    Capt John R Way, Jr                                 *
*       Purpose:    Adaptive Filter and SS Receiver Simulation          *
*       Last Modification: 21 January 1984                              *
*************************************************************************

*************************************************************************
*               VARIABLES AND DECLARATIONS                              *
*************************************************************************
        INTEGER K, KMAX , CASE
        DOUBLE PRECISION X(10000),Y(10000),J(10000),RN(10000),N(10000)
        DOUBLE PRECISION AMP, S(10000)
        DOUBLE PRECISION T, NPWR, PG, FJ, JPWR, TAU
        REAL XK
C  The following are filter variables:
        DOUBLE PRECISION WT(20,1), IDENT(20,20), TERM4(20,1)
        DOUBLE PRECISION A(20,20), AT(20,20), H(20,1), B(20,20)
        DOUBLE PRECISION WC(10), PI, ATB(20,20), ATBA(20,20)
        DOUBLE PRECISION ATBH(20,1)
        DOUBLE PRECISION KS, TERM1(20,20), TERM2(20,1), TERM3(20,1)
        INTEGER Q, L, M, M1, FREQ

        OPEN(15,FILE='results')
        OPEN(16,FILE='arrayres')
        OPEN(17,FILE='wvalues')
        OPEN(18,FILE='h1values')
        OPEN(19,FILE='RNFILE')
        OPEN(13,FILE='h2values')
        OPEN(14,FILE='h3values')
        REWIND 13
        REWIND 14
        REWIND 15
        REWIND 16
        REWIND 17
        REWIND 18
        REWIND 19

*************************************************************************
*          THIS SECTION SETS UP THE INITIAL CONDITIONS                  *
*************************************************************************
        PI = 3.1415926536
        T  = 0.2
        DATA WT/20*0.05/
        DATA (X(Q),Q=1,19)/19*0.05/
        DATA IDENT/400*0.0/
        DATA B/400*0.0/

*************************************************************************
*           THIS LARGE SECTION SETS UP THE PERMANENT ARRAYS             *
*                 FOR THE ADAPTIVE FILTER ALGORITHM.                    *
*************************************************************************

C  THIS SECTION DEFINES THE IDENTITY MATRIX 'IDENT'
        DO 60 Q=1,20
        IDENT(Q,Q)=1.0
```

Figure A-1 ADAPSS Program

```
 60      CONTINUE
         WRITE(16,'(" The following is array IDENT:")')
         CALL MTX9(IDENT,20,20)

C   THIS SECTION DEFINES THE VALUES FOR WC (C FOR CONSTRAINED)
         DO 67 Q=1,6
         WC(Q) =  Q*(2*PI/6.0)
 67      CONTINUE
         WC(7)=10.0
         WC(8)=12.0
         WC(9)=13.0
         WC(10)=14.0
         WRITE(16,'(" The following is vector WC:")')
         CALL VECOUT(WC,10,1)

C   THIS SECTION DEFINES THE VALUES FOR THE VECTOR H
         DO 78 Q=1,19,2
         H(Q,1) = DCOS(TAU*WC((Q+1)/2))
 78      CONTINUE

         DO 82 Q=2,20,2
         H(Q,1) = DSIN(TAU*(-WC(Q/2)))
 82      CONTINUE

         WRITE(16,'(" The following is vector H:")')
         CALL VECOUT(H,20,1)

C THIS SECTION DEFINES THE ARRAY A
         DATA (A(Q,1),Q=1,19,2)/10*1.0/
         DATA (A(Q,1),Q=2,20,2)/10*0.0/

         DO 94 L=1,19,2
         DO 94 Q=2,20
         A(L,Q) = DCOS((Q-1)*WC((L+1)/2)*T)
 94      CONTINUE

         DO 99 L=2,20,2
         DO 99 Q=2,20
         A(L,Q) = DSIN(-(Q-1)*WC(L/2)*T)
 99      CONTINUE
         WRITE(16,'(" The following is array A:")')
         CALL MTX9(A,20,20)

C   THIS SECTION TRANSPOSES THE ARRAY 'A' INTO 'AT'
         DO 107 L=1,20
         DO 107 Q=1,20
         AT(Q,L) = A(L,Q)
 107     CONTINUE
         WRITE(16,'(" The following is array AT:")')
         CALL MTXOUT(AT,20,20)

         PRINT*,'INPUT CASE NUMBER'
         READ*,CASE
         PRINT*,CASE
         WRITE(15,*)
         WRITE(15,'(" THE FOLLOWING RESULTS ARE FOR CASE:",I3)')CASE
         PRINT*,'INPUT THE NUMBER OF FREQUENCIES AT WHICH'
         PRINT*,'THE SINGLE JAMMER WILL APPEAR: 1,2,or3'
         READ*,M
         PRINT*,M
         WRITE(15,'(" NUMBER OF HOPS MADE BY JAMMER:",I2)')M-1
         DO 328 FREQ=1,M

*************************************************************************
```

Figure A-1 ADAPSS Program (cont.)

```
*       THIS SECTION ASKS FOR THE VARIOUS INPUT PARAMETERS:                  *
*               KS    Convergent Factor                                      *
*               KMAX  Number of Iterations plus 19                           *
*               B     Constraint Weighting Matrix                            *
*               JPWR  Jammer Power                                           *
*               FJ    Jammer Frequency                                       *
*               NPWR  Noise Power                                            *
*               PG    Processing Gain                                        *
*************************************************************************
        PRINT*,' INPUT CONVERGENT FACTOR, KS'
        READ*,KS
        PRINT*,KS
        WRITE(15,'(" CONVERGENT FACTOR:",F9.5)')KS
        PRINT*,' INPUT NUMBER OF ITERATIONS + 19'
        READ*,KMAX
        PRINT*,KMAX
        WRITE(15,'(" NUMBER OF ITERATIONS:",I5)')KMAX-19
        PRINT*,' ENTER THE \'B\' ARRAY VALUES BELOW THE '
        PRINT*,' CORRESPONDING CONSTRAINT FREQUENCIES  WC:'
        PRINT*,' 1.Ø 2.1 3.1 4.2 5.2 6.3 1Ø. 12. 13. 14.'
        READ*,B(2,2),B(4,4),B(6,6),B(8,8),B(1Ø,1Ø),B(12,12),B(14,14),
     *       B(16,16),B(18,18),B(2Ø,2Ø)
        WRITE(15,'(" CONSTRAINT WEIGHTS APPEAR BELOW FREQUENCIES")')
        WRITE(15,'(" 1.Ø 2.1 3.1 4.2 5.2 6.3 1Ø. 12. 13. 14.")')
        WRITE(15,9ØØ)(B(Q,Q),Q=2,2Ø,2)
 9ØØ    FORMAT(1Ø(F4.Ø))

        PRINT*,' BEAR IN MIND WHEN SELECTING THE FOLLOWING PARAMETERS'
        PRINT*,' THAT THE POWER OF THE SPREAD SIGNAL IS TAKEN TO BE 1.Ø'
        PRINT*,' AND THE SIGNAL PULSE WIDTH IS ASSUMED TO BE 1.Ø SEC.'
        PRINT*
        WRITE(*,'("INPUT SS RECEIVER\'S PROCESSING GAIN, PG")')
        READ*, PG
        PRINT*,PG
        WRITE(15,'(" PROCESSING GAIN:",F6.Ø)')PG
        WRITE(*,'("INPUT NOISE POWER, NPWR")')
        READ*, NPWR
        PRINT*,NPWR
        WRITE(15,'(" NOISE POWER:",F4.1)')NPWR
        WRITE(*,'("INPUT POWER OF JAMMER, JPWR")')
        READ*, JPWR
        PRINT*,JPWR
        WRITE(15,'(" POWER OF JAMMER:",F5.1)')JPWR
        WRITE(*,'("INPUT FREQUENCY OF JAMMER IN RADIANS, FJ")')
        READ*, FJ
        PRINT*,FJ
        WRITE(15,'(" FREQUENCY OF JAMMER IN RADIANS:",F4.1)')FJ

*************************************************************************
*             THIS SECTION SETS UP THE ARRAYS THAT                      *
*               VARY WITH THE CHOICE OF B VALUES                        *
*************************************************************************

C  THIS SECTION CONTINUES TO DEFINE THE CONSTRAINT MATRIX 'B'
        DO 18Ø Q=1,19,2
        B(Q,Q) = B(Q+1,Q+1)
 18Ø    CONTINUE
        WRITE(16,'(" The following is array B:")')
        CALL BOUT(B,2Ø,2Ø)

C  THIS SECTION COMPUTES:    'AT' TIMES 'B' = 'ATB'
        CALL MULT(AT,B,ATB,2Ø,2Ø,2Ø)
        WRITE(16,'(" The following is array ATB:")')
        CALL MTXOUT(ATB,2Ø,2Ø)
```

Figure A-1 ADAPSS Program (cont.)

```
C  THIS SECTION COMPUTES:    'ATB' TIMES 'A' = 'ATBA'
        CALL MULT(ATB,A,ATBA,2Ø,2Ø,2Ø)
        WRITE(16,'(" The following is array ATBA:")')
        CALL MTX9(ATBA,2Ø,2Ø)

C  THIS SECTION COMPUTES TERM1 =  I + 2*ks*AT*B*A
        DO 19Ø Q=1,2Ø
        DO 19Ø L=1,2Ø
        TERM1(Q,L)=IDENT(Q,L)+2.Ø*KS*ATBA(Q,L)
 19Ø    CONTINUE
        WRITE(16,'(" The following is array TERM1:")')
        CALL MTX9(TERM1,2Ø,2Ø)

C  THIS SECTION COMPUTES:    'ATB' TIMES 'H' = 'ATBH'
        CALL MULT(ATB,H,ATBH,2Ø,2Ø,1)
        WRITE(16,'(" The following is vector ATBH:")')
        CALL VECOUT(ATBH,2Ø,1)

C  THIS SECTION COMPUTES TERM2 = 2*ks*AT*B*H
        DO 192 Q=1,2Ø
        TERM2(Q,1) = 2*KS*ATBH(Q,1)
 192    CONTINUE
        WRITE(16,'(" The following is vector TERM2:")')
        CALL VECOUT(TERM2,2Ø,1)

*****************************************************************
        RN(19) = Ø.Ø
        RN(18) = Ø.Ø
        N(19) = Ø.Ø
        N(18) = Ø.Ø
        S(19) = -1.Ø

C   THE NEXT LINE COMPUTES AMPLITUDE, AMP, OF THE NOISE SPECTRAL
C   SHAPING FILTER WHICH IS NECESSARY TO COMPUTE N(K)
        AMP = DSQRT(NPWR/Ø.17)

C   Start do loop here
        DO 292 K = 2Ø, KMAX

C   Read in rn(k)
        READ(19,*) RN(K)

C   Calculate j(k)
        J(K) = DSQRT(2*JPWR)*DCOS(FJ*K*T)

C   Calculate n(k)
        N(K) = (9*AMP*T**2*(RN(K)+2*RN(K-1)+RN(K-2))-(18*T**2-8)*N(K-1)
     *        - (9*T**2-8.48528*T+4)*N(K-2))/(9*T**2+8.48528*T+4)

C   Calculate s(k)
        XK = K
        IF (MOD(XK,(1./T)).EQ.Ø.Ø) THEN
                IF (RN(K).GT.Ø.Ø) THEN
                S(K) = 1.Ø
                ELSE
                S(K) = -1.Ø
                END IF
                ELSE
                S(K) = S(K-1)
        END IF

C   Calculate x(k)
        X(K) = S(K) + J(K) + N(K)
```

Figure A-1 ADAPSS Program (cont.)

```
*************************************************************************
*          THIS SECTION OF THE PROGRAM WILL COMPUTE THE WEIGHT          *
*               COEFFICIENTS FOR THE ADAPTIVE FILTER.                   *
*************************************************************************

        CALL MULT(TERM1,WT,TERM3,20,20,1)

        Y(K) = X(K)*WT(1,1) + X(K-1)*WT(2,1)+ X(K-2)*WT(3,1)
     *     + X(K-3)*WT(4,1) + X(K-4)*WT(5,1)+ X(K-5)*WT(6,1)
     *     + X(K-6)*WT(7,1) + X(K-7)*WT(8,1)+ X(K-8)*WT(9,1)
     *     + X(K-9)*WT(10,1) + X(K-10)*WT(11,1)+ X(K-11)*WT(12,1)
     *     + X(K-12)*WT(13,1) + X(K-13)*WT(14,1)+ X(K-14)*WT(15,1)
     *     + X(K-15)*WT(16,1) + X(K-16)*WT(17,1)+ X(K-17)*WT(18,1)
     *     + X(K-18)*WT(19,1) + X(K-19)*WT(20,1)

        DO 252 Q=1,20
        TERM4(Q,1) = 2*KS*Y(K)*X(K+1-Q)
 252    CONTINUE

        DO 256 Q=1,20
        WT(Q,1) = TERM3(Q,1) + TERM4(Q,1) - TERM2(Q,1)
 256    CONTINUE

*************************************************************************
*          THIS SECTION DISPLAYS INTERMIDIATE FILTER WEIGHTS            *
*               EVERY 'SHOW' ITERATIONS OF THE DO LOOP                  *
*************************************************************************
C       WHEN RUNNING THE PROGRAM FOR A LARGE NUMBER OF ITERATIONS
C       IT MAY BE DESIRED TO SEE INTERMIDIATE FILTER WEIGHTS.  IF
C       SO, REPLACE 'SHOW' IN THE STATEMENT BELOW WITH THE DESIRED
C       FREQUENCY OF OBSERVATIONS AND REMOVE THE COMMENT CHARACTER
C       'C' FROM THE FOLLOWING FORTRAN STATEMENTS:
C
C       XK=K
C       IF( MOD(XK,50.0).EQ. 0.0 ) THEN
C       PRINT*,' FOR K=',K
C       PRINT*,' WT(1) = ', WT(1,1)
C       PRINT*,' WT(2) = ', WT(2,1)
C       PRINT*,' WT(3) = ', WT(3,1)
C       PRINT*,' WT(4) = ', WT(4,1)
C       PRINT*,' WT(5) = ', WT(5,1)
C       PRINT*,' WT(6) = ', WT(6,1)
C       PRINT*,' WT(7) = ', WT(7,1)
C       PRINT*,' WT(8) = ', WT(8,1)
C       PRINT*,' WT(9) = ', WT(9,1)
C       PRINT*,' WT(10) = ', WT(10,1)
C       PRINT*,' WT(11) = ', WT(11,1)
C       PRINT*,' WT(12) = ', WT(12,1)
C       PRINT*,' WT(13) = ', WT(13,1)
C       PRINT*,' WT(14) = ', WT(14,1)
C       PRINT*,' WT(15) = ', WT(15,1)
C       PRINT*,' WT(16) = ', WT(16,1)
C       PRINT*,' WT(17) = ', WT(17,1)
C       PRINT*,' WT(18) = ', WT(18,1)
C       PRINT*,' WT(19) = ', WT(19,1)
C       PRINT*,' WT(20) = ', WT(20,1)
C       END IF
  292   CONTINUE

        K=K-1
        PRINT*,' FOR K= ', K
        PRINT*,' RN(K)= ', RN(K)
        PRINT*,' S(K) = ', S(K)
        PRINT*,' J(K) = ', J(K)
```

Figure A-1 ADAPSS Program (cont.)

```
      PRINT*,' N(K) = ', N(K)
      PRINT*,' X(K) = ', X(K)
      PRINT*,' WT(1) = ', WT(1,1)
      PRINT*,' WT(2) = ', WT(2,1)
      PRINT*,' WT(3) = ', WT(3,1)
      PRINT*,' WT(4) = ', WT(4,1)
      PRINT*,' WT(5) = ', WT(5,1)
      PRINT*,' WT(6) = ', WT(6,1)
      PRINT*,' WT(7) = ', WT(7,1)
      PRINT*,' WT(8) = ', WT(8,1)
      PRINT*,' WT(9) = ', WT(9,1)
      PRINT*,' WT(10) = ', WT(10,1)
      PRINT*,' WT(11) = ', WT(11,1)
      PRINT*,' WT(12) = ', WT(12,1)
      PRINT*,' WT(13) = ', WT(13,1)
      PRINT*,' WT(14) = ', WT(14,1)
      PRINT*,' WT(15) = ', WT(15,1)
      PRINT*,' WT(16) = ', WT(16,1)
      PRINT*,' WT(17) = ', WT(17,1)
      PRINT*,' WT(18) = ', WT(18,1)
      PRINT*,' WT(19) = ', WT(19,1)
      PRINT*,' WT(20) = ', WT(20,1)

      DO 324 M1=1,19
      X(M1) = X(KMAX-19+M1)
324   CONTINUE

      CALL PWROUT(WT,JPWR,FJ,AMP,PG,T)
      CALL XFER(WT,T,FREQ)
328   CONTINUE

      END

      SUBROUTINE XFER(WT,T,FREQ)
      COMPLEX HXFER,I,HXFER1,HXFER2
      DOUBLE PRECISION WT(20,1)
      DOUBLE PRECISION W, T, ABSH, DELW, DB
      INTEGER L, FREQ
      COMPLEX LPD,I
      REAL PI,A1,A2,A3,A4,A5,A6,A7,A8,A9,PI2
      REAL LPNR,LPDR,LPH
      INTEGER L
      WRITE(15,'(" FOR JAMMER HOP NUMBER:",I2)')FREQ-1
      WRITE(15,'(" THE FREQUENCY RESPONSE IS GIVEN BY:")')
      PI = 3.14159265
      PI2 = PI*2
      A1 = 6.3924532
      A2 = 20.4317291
      A3 = 42.8020611
      A4 = 64.8823963
      A5 = 74.2334292
      A6 = 64.8823963
      A7 = 42.8020611
      A8 = 20.4317291
      A9 = 6.3924532
      I = (0.0,1.0)
      DELW = 0.04
      DO 396 L= 1, 200
      W = L*DELW
      HXFER1 = WT(1,1) * (DCOS(19*W*T) + I*(DSIN(19*W*T)))
     *       +  WT(2,1) * (DCOS(18*W*T) + I*(DSIN(18*W*T)))
     *       +  WT(3,1) * (DCOS(17*W*T) + I*(DSIN(17*W*T)))
     *       +  WT(4,1) * (DCOS(16*W*T) + I*(DSIN(16*W*T)))
     *       +  WT(5,1) * (DCOS(15*W*T) + I*(DSIN(15*W*T)))
```

Figure A-1 ADAPSS Program (cont.)

```
     *       +  WT(6,1) * (DCOS(14*W*T) + I*(DSIN(14*W*T)))
     *       +  WT(7,1) * (DCOS(13*W*T) + I*(DSIN(13*W*T)))
     *       +  WT(8,1) * (DCOS(12*W*T) + I*(DSIN(12*W*T)))
     *       +  WT(9,1) * (DCOS(11*W*T) + I*(DSIN(11*W*T)))
     *       +  WT(10,1) * (DCOS(10*W*T) + I*(DSIN(10*W*T)))
          HXFER2 = WT(11,1) * (DCOS(9*W*T) + I*(DSIN(9*W*T)))
     *       +  WT(12,1) * (DCOS(8*W*T) + I*(DSIN(8*W*T)))
     *       +  WT(13,1) * (DCOS(7*W*T) + I*(DSIN(7*W*T)))
     *       +  WT(14,1) * (DCOS(6*W*T) + I*(DSIN(6*W*T)))
     *       +  WT(15,1) * (DCOS(5*W*T) + I*(DSIN(5*W*T)))
     *       +  WT(16,1) * (DCOS(4*W*T) + I*(DSIN(4*W*T)))
     *       +  WT(17,1) * (DCOS(3*W*T) + I*(DSIN(3*W*T)))
     *       +  WT(18,1) * (DCOS(2*W*T) + I*(DSIN(2*W*T)))
     *       +  WT(19,1) * (DCOS(1*W*T) + I*(DSIN(1*W*T)))
     *       +  WT(20,1)
          HXFER = HXFER1 + HXFER2
          ABSH = CABS(HXFER)
          LPNR = (2*PI)**10
          LPD=-W**10 + A8*W**8*PI2**2 - A6*W**6*PI2**4 + A4*W**4*PI2**6
     *     - A2*W**2*PI2**8 + PI2**10 + I*(A9*W**9*PI2 - A7*W**7*PI2**3
     *     + A5*W**5*PI2**5 - A3*W**3*PI2**7 +A1*W*PI2**9)
          LPDR = CABS(LPD)
          LPH = LPNR/LPDR
          DB = 20*DLOG(ABSH*LPH)
          IF(FREQ.EQ.1)THEN
               WRITE(17,*)W
               WRITE(18,*)DB
             ELSE IF (FREQ.EQ.2) THEN
               WRITE(13,*)DB
                 ELSE
               WRITE(14,*)DB
          END IF
          WRITE(15,'(" W = ",F5.2,T14," H(W) = ",E12.5)')W,DB
 396      CONTINUE
          END

C    THIS SUBROUTINE PRINTS OUT ARRAYS WITH 12 COLUMNS
          SUBROUTINE MTXOUT(AR,ROWS,COLS)
          INTEGER J,L,ROWS,COLS
          DOUBLE PRECISION AR(ROWS,COLS)
          DO 405 J=1,ROWS
          WRITE(16,901)(AR(J,L),L=1,COLS)
 405      CONTINUE
          WRITE(16,*)
 901      FORMAT (12(F6.3))
          RETURN
          END

C    THIS SUBROUTINE PRINTS OUT THE B ARRAY
          SUBROUTINE BOUT(AR,ROWS,COLS)
          INTEGER J,L,ROWS,COLS
          DOUBLE PRECISION AR(ROWS,COLS)
          DO 417 J=1,ROWS
          WRITE(16,902)(AR(J,L),L=1,COLS)
 417      CONTINUE
          WRITE(16,*)
 902      FORMAT (20(F4.1))
          RETURN
          END

C    THIS SUBROUTINE PRINTS OUT ARRAYS WITH 9 COLUMNS
          SUBROUTINE MTX9(AR,ROWS,COLS)
          INTEGER J,L,ROWS,COLS
          DOUBLE PRECISION AR(ROWS,COLS)
```

Figure A-1 ADAPSS Program (cont.)

```
      DO 429 J=1,ROWS
      WRITE(16,9Ø3)(AR(J,L),L=1,COLS)
 429  CONTINUE
      WRITE(16,*)
 9Ø3  FORMAT (9(1X, F7.4))
      RETURN
      END

C  THIS SUBROUTINE PRINTS OUT VECTORS
      SUBROUTINE VECOUT(AR,ROWS,COLS)
      INTEGER J,L,ROWS,COLS
      DOUBLE PRECISION AR(ROWS,COLS)
      DO 441 J=1,ROWS
      WRITE(16,9Ø4)(AR(J,L),L=1,COLS)
 441  CONTINUE
      WRITE(16,*)
 9Ø4  FORMAT ( F12.9)
      RETURN
      END

C  THIS SUBROUTINE MULTIPLYS A1 BY B1 TO GET C1
      SUBROUTINE MULT(A1,B1,C1,AROW,ACOL,BCOL)
      INTEGER J,L,K,AROW,ACOL,BCOL
      DOUBLE PRECISION SUM,A1(AROW,ACOL),B1(ACOL,BCOL),C1(AROW,BCOL)
      DO 458 K=1,BCOL
      DO 458 L=1,AROW
      SUM=Ø.Ø
      DO 456 J=1,ACOL
      SUM=SUM+A1(L,J)*B1(J,K)
 456  CONTINUE
      C1(L,K)=SUM
 458  CONTINUE
      RETURN
      END


      SUBROUTINE PWROUT(WT,JPWR,FJ,AMP,PG,T)
C     This section computes the powers at the filter output
C     regardless of how many iterations are performed to
C     obtain the filter weights
      INTEGER K
      DOUBLE PRECISION WT(2Ø,1),JPWR,FJ,AMP,PG,T
      REAL S(Ø:221), J(221), N(-1:221), RN(-1:221), V(221)
      REAL YHAT,NOPOUT,SIPOUT,JAPOUT,PWR,NCALC,NOISIN,XK
      RN(Ø)=Ø.Ø
      RN(-1)=Ø.Ø
      N(Ø)=Ø.Ø
      N(-1)=Ø.Ø
      S(Ø)=-1.Ø

C start do loop here
      DO 5Ø4 K = 1, 22Ø

C     Read in rn(k)
      READ(19,*) RN(K)

C     Calculate j(k)
      J(K) = DSQRT(2*JPWR)*DCOS(FJ*K*T)

C     Calculate n(k)
      N(K) = (9*AMP*T**2*(RN(K)+2*RN(K-1)+RN(K-2))-(18*T**2-8)*N(K-1)
     *       - (9*T**2-8.48528*T+4)*N(K-2))/(9*T**2+8.48528*T+4)

C     Calculate the noise input power
```

Figure A-1 ADAPSS Program (cont.)

```
        NCALC = NCALC + N(K)**2

C   Calculate s(k)
        XK = K
        IF (MOD(XK,(1./T)).EQ.0.0) THEN
                IF (RN(K).GT.0.0) THEN
                S(K) = 1.0
                ELSE
                S(K) = -1.0
                END IF
                ELSE
                S(K) = S(K-1)
        END IF
 504    CONTINUE

        PWR = 0.0
        YHAT = 0.0
        DO 518 K=20,219
        V(K) = S(K)
        YHAT = V(K)*WT(1,1) + V(K-1)*WT(2,1)+ V(K-2)*WT(3,1)
     *     + V(K-3)*WT(4,1) + V(K-4)*WT(5,1)+ V(K-5)*WT(6,1)
     *     + V(K-6)*WT(7,1) + V(K-7)*WT(8,1)+ V(K-8)*WT(9,1)
     *     + V(K-9)*WT(10,1) + V(K-10)*WT(11,1)+ V(K-11)*WT(12,1)
     *     + V(K-12)*WT(13,1) + V(K-13)*WT(14,1)+ V(K-14)*WT(15,1)
     *     + V(K-15)*WT(16,1) + V(K-16)*WT(17,1)+ V(K-17)*WT(18,1)
     *     + V(K-18)*WT(19,1) + V(K-19)*WT(20,1)
        PWR = PWR + YHAT**2
 518    CONTINUE
        SIPOUT = PWR/200.0
        WRITE(15,'(" SIGNAL POWER AT FILTER OUTPUT: ",E14.6)')SIPOUT

        PWR = 0.0
        YHAT = 0.0
        DO 534 K=20,219
        V(K) = J(K)
        YHAT = V(K)*WT(1,1) + V(K-1)*WT(2,1)+ V(K-2)*WT(3,1)
     *     + V(K-3)*WT(4,1) + V(K-4)*WT(5,1)+ V(K-5)*WT(6,1)
     *     + V(K-6)*WT(7,1) + V(K-7)*WT(8,1)+ V(K-8)*WT(9,1)
     *     + V(K-9)*WT(10,1) + V(K-10)*WT(11,1)+ V(K-11)*WT(12,1)
     *     + V(K-12)*WT(13,1) + V(K-13)*WT(14,1)+ V(K-14)*WT(15,1)
     *     + V(K-15)*WT(16,1) + V(K-16)*WT(17,1)+ V(K-17)*WT(18,1)
     *     + V(K-18)*WT(19,1) + V(K-19)*WT(20,1)
        PWR = PWR + YHAT**2
 534    CONTINUE
        JAPOUT = PWR/200.0
        WRITE(15,'(" JAMMER POWER AT FILTER OUTPUT: ",E14.6)')JAPOUT

        PWR = 0.0
        YHAT = 0.0
        DO 550 K=20,219
        V(K) = N(K)
        YHAT = V(K)*WT(1,1) + V(K-1)*WT(2,1)+ V(K-2)*WT(3,1)
     *     + V(K-3)*WT(4,1) + V(K-4)*WT(5,1)+ V(K-5)*WT(6,1)
     *     + V(K-6)*WT(7,1) + V(K-7)*WT(8,1)+ V(K-8)*WT(9,1)
     *     + V(K-9)*WT(10,1) + V(K-10)*WT(11,1)+ V(K-11)*WT(12,1)
     *     + V(K-12)*WT(13,1) + V(K-13)*WT(14,1)+ V(K-14)*WT(15,1)
     *     + V(K-15)*WT(16,1) + V(K-16)*WT(17,1)+ V(K-17)*WT(18,1)
     *     + V(K-18)*WT(19,1) + V(K-19)*WT(20,1)
        PWR = PWR + YHAT**2
 550    CONTINUE
        NOPOUT = PWR/200.0
        WRITE(15,'(" NOISE POWER AT FILTER OUTPUT: ",E14.6)')NOPOUT
        NOISIN = NCALC/220.0
        WRITE(15,'(" CALCULATED INPUT NOISE POWER: ",E14.6)')NOISIN
```

Figure A-1 ADAPSS Program (cont.)

```
      WRITE(15,*)
      WRITE(15,'(" IF THE ADAPTIVE FILTER WAS NOT USED, THE")')
      WRITE(15,'(" OUTPUT RESULTS OF THE SS RECEIVER WOULD BE:")')
      WRITE(15,'(" S/N= ",E14.6)') PG/NOISIN
      WRITE(15,'(" S/J= ",E14.6)') PG/JPWR
      WRITE(15,'(" S/(N+J)= ",E14.6)') PG/(NOISIN+JPWR)
      WRITE(15,*)
      WRITE(15,'(" IF THE ADAPTIVE FILTER IS USED, THE OUTPUT")')
      WRITE(15,'(" RESULTS OF THE SS RECEIVER WOULD BE:")')
      WRITE(15,'(" S/N= ",E14.6)') PG*SIPOUT/NOPOUT
      WRITE(15,'(" S/J= ",E14.6)') PG*SIPOUT/JAPOUT
      WRITE(15,'(" S/(N+J)= ",E14.6)') PG*SIPOUT/(NOPOUT+JAPOUT)
      RETURN
      END
```

Figure A-1 ADAPSS Program (cont.)

Appendix B

PSD Program

The PSD program that follows was written to find the power spectral density (PSD) of the SS signal and noise at the input of the adaptive filter. This program calculates the SS signal and noise in the same manner as the ADAPSS program except that 1024 sample points are computed. Then the PSDs are found by utilizing a decimation-in-frequency FFT subroutine described by Oppenheim and Schafer (Ref. 44:331). The values of the PSDs are stored in files for subsequent plotting.

Input File

RNFILE - This file of random numbers with a normal (0,1) distribution must reside in memory before execution of the PSD Program begins. The generation of this file is covered in the "Preliminary Steps" section of chapter IV.

Output Files

wvalues - frequency values in radians used to plot the PSD after completion of the program.

svalues - PSD values of the signal corresponding to the frequencies in the wvalues file.

nvalues - PSD values of the noise.

results - a general file containing various parameters of the results.

Integer Variables

K - main do loop counter.

KMAX - value of final do loop iteration.

Q - do loop counter.

M - value required by FFT subroutine. Total number of data points = 2**M.

Real Variables

RN - random number read from RNFILE.

N - noise value.

S - spread spectrum signal value.

AMP - amplitude of noise spectral shaping filter.

T - sampling time.

XK - the integer K converted to a real number.

NPWR - noise power.

W - frequency in radians.

PI - 3.1415926536.

PSS - PSD of the signal.

PSN - PSD of the noise.

Complex Variables

I - imaginary number (0+j1).

SCOM - complex value of SS signal.

NCOM - complex value of noise

```
      PROGRAM PSD
C  This program calculates the P.S.D. of the
C  inputs S(K) and N(K) for subsequent plotting
*****************************************************************
*                    VARIABLES AND DECLARATIONS                 *
*****************************************************************
      INTEGER K, KMAX, M, Q
      PARAMETER (KMAX = 1100)
      REAL RN(-2:KMAX), N(-2:KMAX), S(-2:KMAX), AMP
      REAL T, XK, NPWR, W, PI
      REAL PSS(0:511), PSN(0:511)
      COMPLEX I, SCOM(0:KMAX),NCOM(0:KMAX)
      OPEN(15,FILE='results')
      OPEN(16,FILE='RNFILE')
      OPEN(17,FILE='wvalues')
      OPEN(18,FILE='svalues')
      OPEN(19,FILE='nvalues')
      REWIND 15
      REWIND 16
      REWIND 17
      REWIND 18
      REWIND 19
*****************************************************************
*           THIS SECTION SETS UP THE INITIAL CONDITIONS         *
*****************************************************************
      WRITE(*,'("INPUT NOISE POWER, NPWR")')
      READ*, NPWR
      PRINT*,NPWR
      PI = 3.1415926536
      T  = 0.2
      M = 10
      RN(-1) = 0.0
      N(-1) = 0.0
      S(-1) = 1.0
      RN(-2) = 0.0
      N(-2) = 0.0
      S(-2) = 1.0
      I=(1.0,0.0)
*****************************************************************
C  Compute amplitude of the noise spectral shaping filter
      AMP = SQRT(NPWR/0.1)
C  Start do loop
      DO 59 K = 0, KMAX
C  Obtain rn(k)
      READ(16,*) RN(K)
C  Calculate n(k)
      N(K)=(9*AMP*T**2*(RN(K)+2*RN(K-1)+RN(K-2))-(18*T**2-8)*N(K-1)
     *     -(9*T**2-8.48528*T+4)*N(K-2))/(9*T**2+8.48528*T+4)
C  Calculate s(k)
      XK = K
      IF (MOD(XK,(1.0/T)).EQ.0.0) THEN
            IF (RN(K).GT.0.0) THEN
                  S(K) = 1.0
                  ELSE
                  S(K) = -1.0
            END IF
                  ELSE
            S(K) = S(K-1)
      END IF
C  Convert S(K) and N(K) to complex
      SCOM(K) = CMPLX(S(K))
      NCOM(K) = CMPLX(N(K))
C  End do loop
 59   CONTINUE
```

Figure B-1 PSD Program

```
      WRITE(15,'(" For Noise Power = ",F6.3)')NPWR
      WRITE(15,'(" The Results of the PSD Program are:")')
C  Since there are 1024 data points, the intervals will be
C  2*PI/1024T=0.030679616 rad/sec, therefore, w = 0.030679616*Q
C  Calculate P.S.D. of S(K)
      CALL FFT(SCOM,M)
      DO 72 Q=0,260
      PSS(Q) = ((CABS(SCOM(Q)))**2)/1024.
      W=Q*0.030679616
      WRITE(15,'(" For w =",F8.4,"  P.S.D. of S =",F12.8)')W,PSS(Q)
      WRITE(17,*)W
      WRITE(18,*)PSS(Q)
 72   CONTINUE
C  Calculate P.S.D. of N(K)
      CALL FFT(NCOM,M)
      DO 83 Q=0,260
      PSN(Q) = (CABS(NCOM(Q))**2)/1024.
      W=Q*0.030679616
      WRITE(15,'(" For w =",F8.4,"  P.S.D. of N =",F12.8)')W,PSN(Q)
      WRITE(19,*)PSN(Q)
 83   CONTINUE
      END

      SUBROUTINE FFT(X,M)
C  This subroutine calculates the Fast Fourier Transform
C  from the 2**M complex values of the vector X
      COMPLEX X(1024),U,W,T
      INTEGER I,J,K,N,L,LE,LE1,IP,NV2,NM1,M
      REAL PI
      N=2**M
      PI=3.14159265358979
      DO 20 L=1,M
      LE=2**(M+1-L)
      LE1=LE/2
      U=(1.0,0.0)
      W=CMPLX(COS(PI/FLOAT(LE1)),-SIN(PI/FLOAT(LE1)))
      DO 20 J=1,LE1
      DO 10 I=J,N,LE
      IP=I+LE1
      T=X(I)+X(IP)
      X(IP)=(X(I)-X(IP))*U
 10   X(I)=T
 20   U=U*W
      NV2=N/2
      NM1=N-1
      J=1
      DO 30 I=1,NM1
      IF(I.GE.J) GO TO 25
      T=X(J)
      X(J)=X(I)
      X(I)=T
 25   K=NV2
 26   IF(K.GE.J) GO TO 30
      J=J-K
      K=K/2
      GO TO 26
 30   J=J+K
      RETURN
      END
```

Figure B-1 PSD Program (cont.)

Vita

John Robert Way, Jr., was born on 6 October 1955 in Charleston, South Carolina. He graduated from high school in Redlands, California in 1974 and attended North Carolina State University from which he received the degree of Bachelor of Science in Electrical Engineering in May 1978. Upon graduation, he received a commission in the USAF through the ROTC program. He immediately began active duty and served first as Titan III Booster Systems Engineer and then as Titan III-D Satellite Engineer with the 6595th Aerospace Test Group, Vandenberg AFB, California. While at Vandenberg AFB, he earned the degree of Master of Science in Systems Management from the University of Southern California through part time study. He entered the School of Engineering, Air Force Institute of Technology, in June 1982.

Permanent address: 626 Maxine St.

Fayetteville, North Carolina 28303

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1. REPORT SECURITY CLASSIFICATION: UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S): AFIT/GE/EE/84S-12	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION: School of Engineering; 6b. OFFICE SYMBOL (If applicable): AFIT/EN	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State and ZIP Code): Air Force Institute of Technology, Wright-Patterson AFB, Ohio 45433	7b. ADDRESS (City, State and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION; 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State and ZIP Code)	10. SOURCE OF FUNDING NOS.

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT NO.

11. TITLE (Include Security Classification): See Box 19

12. PERSONAL AUTHOR(S): John R. Way, Jr., Capt, USAF

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Yr., Mo., Day)	15. PAGE COUNT
MS Thesis	FROM _____ TO _____	1984 June	118

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB. GR.	Soft-Constraint LMS Algorithm, Adaptive Filter, LMS Algorithm, Spread Spectrum, Digital Communication
17	02		

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Title: THE PERFORMANCE OF A NEW ADAPTIVE FILTER FOR DIGITAL COMMUNICATIONS

Thesis Advisor: Dr. V.H. Syed

Approved for public release: IAW AFR 190-17.
LYNN E. WOLAVER
Dean for Research and Professional Development
Air Force Institute of Technology (AIC)
Wright-Patterson AFB OH 45433

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS ☐	UNCLASSIFIED

22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE NUMBER (Include Area Code)	22c. OFFICE SYMBOL
V. H. Syed, PhD	513-255-4960	AFIT/ENG

DD FORM 1473, 83 APR — EDITION OF 1 JAN 73 IS OBSOLETE.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

The improvement in performance of a digital communications receiver preceded by an adaptive filter using the soft-constraint Least Mean Square algorithm is found. This algorithm has previously only been used in conjunction with adaptive antenna arrays. The fundamentals of adaptive filters and the spread spectrum technique are presented. A computer model is developed which simulates a 19 tap adaptive filter and spread spectrum receiver combination in the baseband region. The filter is shown to adapt well to hopping jammers and dual jammers. Plots of the filter's transfer function show notch depths ranging from -34 dB to -77 dB over iterations ranging from 15 to 5000 for the case of a 20 db sinusoid jammer. The improvement gained by adjusting the softness of the constraints is presented. The computer program and recommendations for further study are included as well as a 44 item bibliography.

ORIGINATOR - SUPPLIED KEY WORDS INCLUDE:

END

FILMED

4-85

DTIC